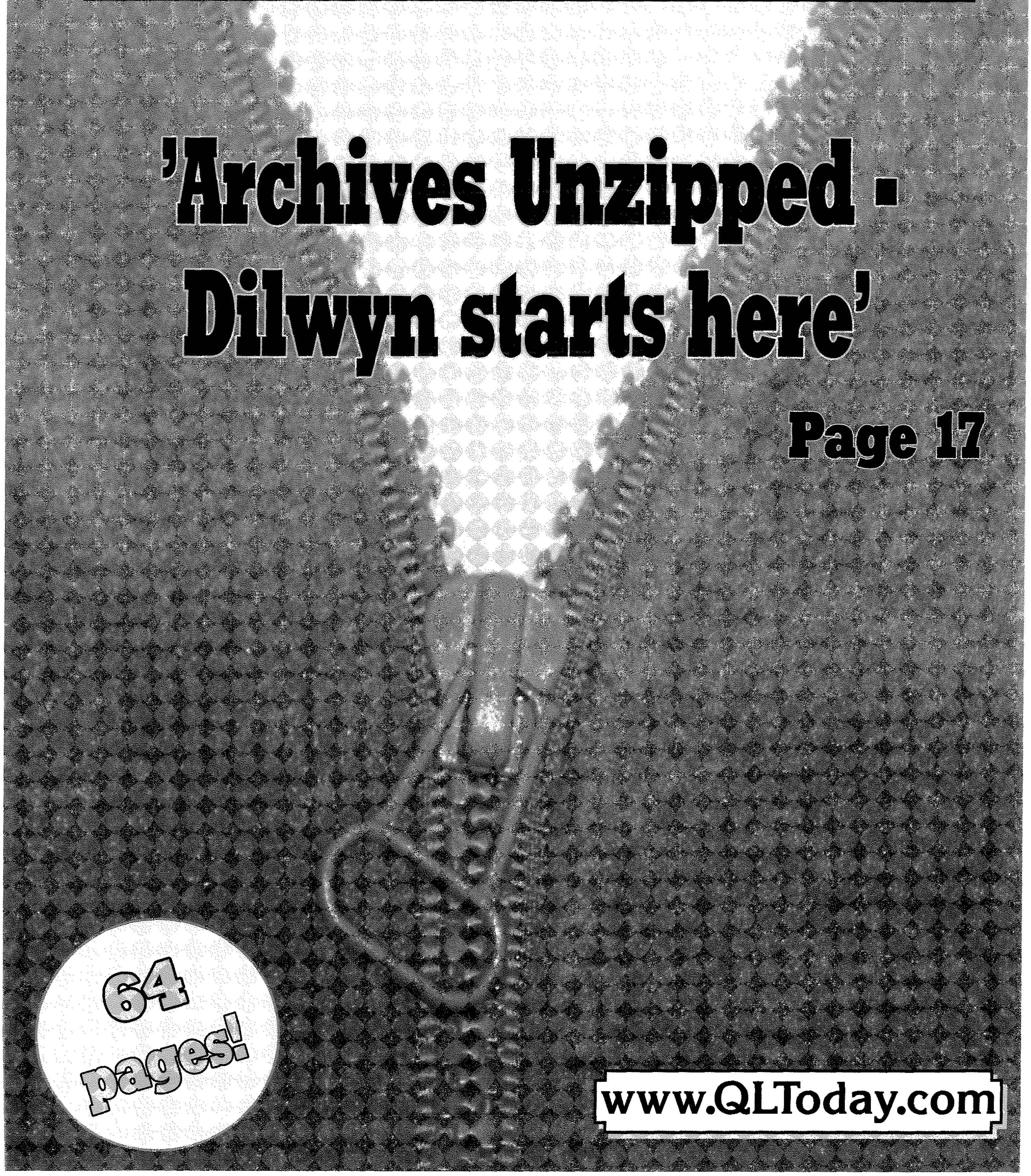
QL Today

Volume 11
Issue 4
March-May
2007

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...



'Archives Unzipped - Dilwyn starts here'

Page 17

64 pages!

www.QLToday.com

Contents

3 Editorial

4 News

10 Enigma — *Ian Pine*

15 Using QPC under Linux: an example — *Wolfgang Lenerz*

17 ZIP and UNZIP — *Dilwyn Jones*

25 Co-Prozessing — *Stephen Poole*

26 Monday Morning QL'ing Guide — *Dilwyn Jones*

30 Compiling QPTR with TURBO — *George Gwilt*

44 GPS and QPC - Part 3 — *Hugh Rooms*

52 Small Ads

53 Code Timer - Benchmarks — *Stephen Poole*

55 Programming Assembler - Part 18 — *Norman Dunbar*

57 Byts of Wood — *Roy Wood*

62 A personal Statement from the Editor — *Geoff Wicks*

63 Changes Ahead — *Geoff Wicks*

Advertisers

in alphabetical order

Jochen Merz Software 9
QBranch 34, 35
Quanta 19
RWAP 61
TF Services 27
Geoff Wicks 43

QL *Today*

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software	Tel. +49 203 502011
Kaiser-Wilhelm-Str. 302	Fax +49 203 502012
47169 Duisburg	email: JMerz@j-m-s.com
Germany	email: QLToday@j-m-s.com

English office:

Q Branch	Tel. +44 1273 386030
20 Locks Hill	Mobile +44 7836 745501
Portslade	Fax +44 1273 381577
BN41 2LB	email: qbranch@qbranch.demon.co.uk
United Kingdom	email: QLToday@j-m-s.com

Editor:

Geoff Wicks	Tel. +44 1332 271366
5b Wordsworth Avenue	email: gwicks@beeb.net
Sinfin	email: QLToday@j-m-s.com
Derby DE24 9HQ	
United Kingdom	

Co-Editor:

Bruce Nicholls	Tel +44 20 71930539
38 Derham Gardens	Fax +44 870 0568755
Upminster	email: qltoday@q-v-d.demon.co.uk
Essex RM14 3HA	email: QLToday@j-m-s.com
United Kingdom	

QL *Today* is published five times a year, our volume begins on beginning of June. Please contact the German or English office for current subscription rates or visit our homepage www.QLTODAY.com.

We welcome your comments, suggestions and articles. YOU make **QL *Today*** possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF or JPG. To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

If you need more information about the UNZIP program which is used by our BOOT program to unpack the files, we suggest that you visit Jonathan Hudsons web site where you find more information about lots of interesting QDOS software and INFOZIP at www.bigfoot.com/~jrhudson/

The deadline for the next issue is the 15th of May 2007

Editorial

by Geoff Wicks

Zilch. Total zilch. Absolutely nothing happened, and that was what I was expecting.

A few months ago one of our regular writers sent us a pained email. He had, with great enthusiasm, floated an idea in the magazine, but was bitterly disappointed because no one had responded to it.

At QL Today we are more battle-hardened and know that is the normal state of affairs. We received no reaction to our cover challenge in issue 1 and no reaction to our remarks about shows in the last issue. We probably get feedback to about 1 in 10 of the ideas we float in the magazine, but that 1 in 10 reaction is usually detailed and cogent enough to make our efforts worthwhile.

It is easier to get feedback on the magazine as a whole. Last year you gave us feedback we like when all but seven of you renewed your subscriptions. (And some who did not renew are now doing their QL-ing in that big internet cafe in the sky.) Other than at renewal time we get very little direct feedback from our readers, but if we keep our eyes and ears open at shows, on the internet and in other QL publications we can build up a reasonable picture of what you think of us.

Listening is an important part of feedback, and maybe Quanta could learn a thing or two from us.

Last year we reported a serious error in the published Quanta constitution. This stated 100 members had to sign a request for a special general meeting, although only 11 signatures are required. In total I have warned Quanta of this error four times, both personally and in QL Today, but the committee has failed to correct it or apologise to the members.

The Quanta committee has embarked on a policy of increasing their own power, but reducing that of the members. Two years ago the present officers voted for a three year term of office instead of one. Last year at the AGM the chairman ruled there could be no "Any Other Business", but only an "Open Discussion". This year they are seeking to tighten the restrictions on the right to call a Special General Meeting. If the committee have their way 20 members will have to sign a petition and pay £300 up front.

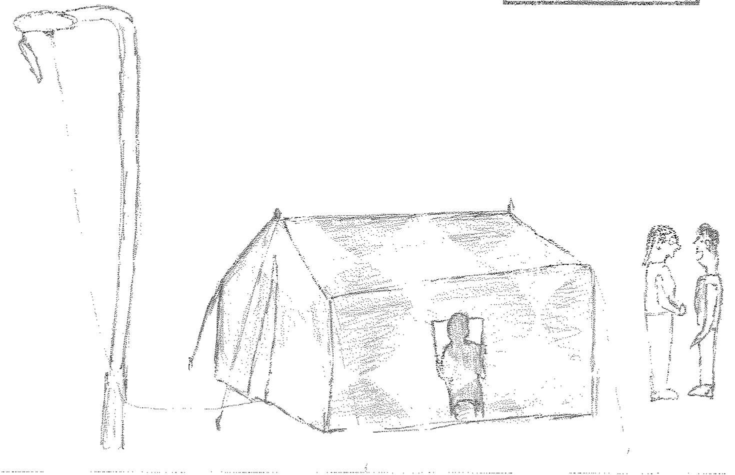
Do you know of any other organisation with a structure that gives so much power to the executive and so little to its members?

An executive with so much power must go out of its way to prove its sensitivity to the opinions, feelings and rights of its members. Saying sorry for misleading them would be a good start.

A tough challenge, but let us not forget more welcome developments. QL Today is impressed by improvements to the Quanta Magazine and we are looking forward to the, shortly promised, renewed Quanta website.

Cartoon

by Roy Wood



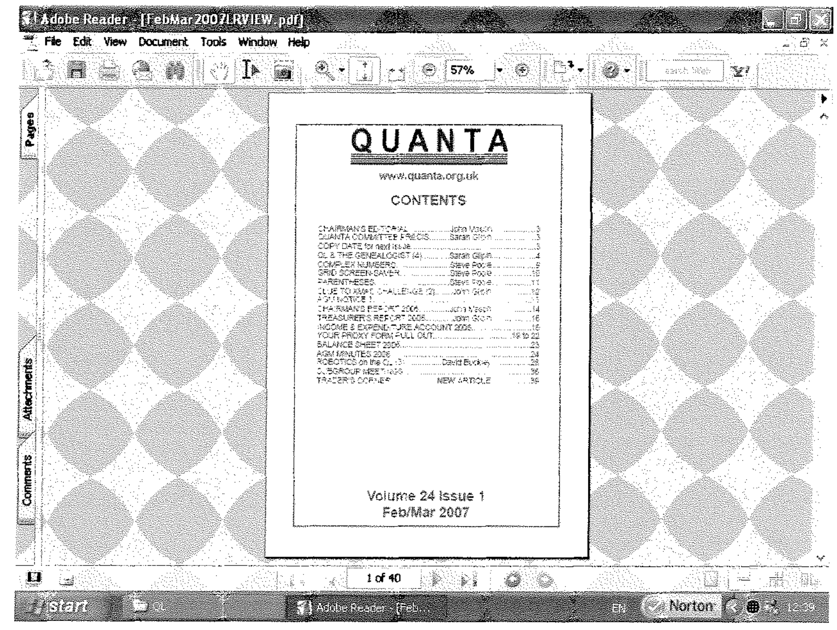
Sad case really. By the time he had bought the hardware to be able to run Vista he could not pay his mortgage!

QUANTA First

Quanta has published the first electronic edition of the Quanta Magazine available to all members. It is a bumper 40 page issue weighing in at a slim 337Kb. In addition to the full documents for the 2007 AGM it contains articles on Genealogy, SuperBasic programming and robotics. Quanta members can now choose between the electronic or the printed version of the magazine.

Hero of the piece is acting editor John Gilpin. He temporarily took over the editorship in 2005 even though he had no previous experience of magazine publishing. John quickly learnt to master Page Plus and, during his tenure, the magazine has vastly improved in both content and print quality. Praise also has to be given to Quanta chairman, John Mason, for the high priority he has given to the magazine, which previously had been neglected by successive committees.

In addition to his duties as acting editor, John Gilpin is also treasurer, membership secretary, and acting webmaster of Quanta. Subscribers to the QL-user email list also know him as the one member of the Quanta committee who actively strives to keep up to date with QL developments.


Adobe Reader - [FebMar2007LRVIEW.pdf]
File Edit View Document Tools Window Help
57%
Pages
Attachments
Comments
1 of 40
start
QL
Adobe Reader - [Feb...
EN
Norton
12:39

QUANTA

www.quanta.org.uk

CONTENTS

CHAIRMAN'S EDITORIAL .... John Mason 3
QUANTA COMMITTEE PRECIS .... Sarah Gilpin 3
COPY DATE for next issue 3
QL & THE GENEALOGIST (4) .... Sarah Gilpin 4
COMPLEX NUMBERS .... Steve Poole 5
GRID SCREEN-SAVER .... Steve Poole 10
PARENTHESES .... Steve Poole 11
CLUE TO XMAS CHALLENGE (2) .... John Gilpin 12
AGM NOTICE 1 13
CHAIRMAN'S REPORT 2006 .... John Mason 14
TREASURER'S REPORT 2006 .... John Gilpin 16
INCOME & EXPENDITURE ACCOUNT 2006 16
YOUR PROXY FORM PULL OUT .... 18 to 22
BALANCE SHEET 2006 23
AGM MINUTES 2006 24
ROBOTICS on the QL (3) .... David Buckley 28
SUBGROUP MEETINGS 36
TRADER'S CORNER .... NEW ARTICLE 36

Volume 24 Issue 1
Feb/Mar 2007

The magazine has been produced in electronic form for about 18 months, but distribution has been restricted to the Quanta committee and a small group of testers. The biggest problem to be overcome was the file size. Early editions had a file size of about 1Mb, but the committee wanted to get this down to under 500Kb for members without a broadband connection.

The reduction in file size has been achieved at the cost of traders, who have lost their full page adverts, and now have to make do with a few lines of text pushed to the back of the magazine. (In return Quanta provides free advertising to traders who are full members of the organisation.) The traders were promised that their full advertisements would be placed on a new traders' section of the Quanta website and that the electronic magazine would have links to these and to their own websites.

This promise has yet to be kept. The Quanta website appears not to have been updated in over a year and will not be updated before the Quanta AGM.

In a email to traders John Gilpin wrote ominously: *"We hope to launch the new site at the Portslade Workshop/AGM in April, or at least have some development ideas to show and discuss with you".*

A further disappointment for traders was that about half of the hyperlinks in the electronic edition of the magazine did not work, which raised questions about the thoroughness of the testing. John Gilpin quickly responded to traders' comments and discovered the problem was caused by a software quirk in Page Plus' handling of PDF files. The fault has now been corrected.

Biggest loser of all is QL Today. Quanta and QL Today have advertised in each other's publications since the launch of QL Today in 1996 under a reciprocal agreement not to charge for each other's advertisements. Over the years this agreement has been to the disadvantage of QL Today. Although the publishing frequency of QL Today has fallen from 6 to 5 copies per year, that of the Quanta Magazine has gone from 12 to 6 copies. QL Today has lost its display advert in the Quanta Magazine and is now relegated to a third of an A5 page of text at the back of the magazine. Quanta still has 5 A4 display advertisements per year in QL Today.

Mixed News

The Quanta Magazine AGM edition brings with it a mixed bag of news. 2006 brought another sharp decline in Quanta membership, but this was not as severe as in 2005. Income has fallen by over 18% from £4,616 to £3,400. Over the year Quanta made a loss of £618. Workshop costs have risen from £395 to £1,047. The Quanta Committee has broken with a long-standing tradition of publishing membership details in the AGM edition of the magazine, but there was a fall of almost 13% in subscription income, which would suggest a loss of 25 - 30 members during the year.

On the positive side Quanta has now a volunteer to act as webmaster, Dan Abbott, who has been nominated for a committee post. Dan joined Quanta and accepted nomination to the committee to become webmaster. He is hard needed as the Quanta website appears not to have been updated for over a year and is still advertising the next QL show as the 2006 Manchester AGM. Quanta plans a minimum of two workshops per year but in ambiguous wording. The chairman refers to Quanta "holding" these, whereas the Treasurer only to "financially supporting" them. (What happens if no subgroup wants to organise a second show in 2007? Will the Quanta committee then take over and organise it centrally?) The committee received an offer from the Midlands to run the 2007 AGM, but this arrived after the decision to hold it in Hove. There has been no QL workshop in the Midlands since 1998 and a show in this region is well overdue.

On a lighter side Quanta has reduced the value of its stock to reflect its lower realisable value. This includes 54 of the notorious "QL is 21" T-shirts that must now have the status of a collector's item. They are only available in size XL. (Quanta's committee "came out" in a big way at QL is 21.)

EURO Converter Update

Davide Santachiara writes:

"Andrea Carpi just released a new version of Euro converter, a useful tool to convert Euro from/to the former european currencies. The new revision is now GD2 compatible (Q40/Q60 volunteers are welcome as it was tested on QPC2 only) and now includes the Slovenian tolar as Slovenia is going to use Euro on 1.1.2007.

Download from

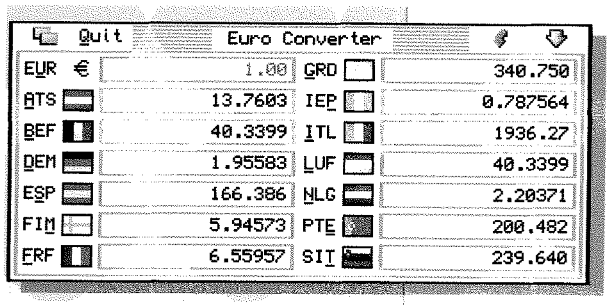
http://it.groups.yahoo.com/group/sinclair-italy/files/

(file area of the italian QL mailing list)

The SBasic program, written by Andrea with Easymenu and QLiberated is freeware"

Some users have reported difficulties in downloading from this site. An alternative is:

http://www.geocities.com/dsantachiara/qlpage.htm



```
   Quit          Euro Converter
EUR €        1.00  GRD       340.750
ATS       13.7603  IEP      0.787564
BEF       40.3399  ITL       1936.27
DEM       1.95583  LUF       40.3399
ESP       166.386  NLG       2.20371
FIM       5.94573  PTE       200.482
FRF       6.55957  SIT       239.640
```

Sound Bug

George Gwilt has spotted a bug in QPC's sound handling:

"The Sampled Sound System (SSS) in SMSQE allows you to send different sounds to the left and right speakers, thus giving the possibility of stereo sound.

This works as expected on the Q40 and Q60 but not on QPC2. In version 3.33 anything sent to the right channel is ignored and anything sent to the left channel goes to both speakers. Thus with QPC2 you get a mono sound.

The SMSQE source code for Q40/60 is very clear and simple. A byte is sent to the left DAC and the next byte is sent to the right DAC. This, of course, is what gives the possibility of stereo. The source code for QPC2 is obviously different - but surely it should allow the right hand speaker to have its say."

Marcel Kilgus replied:

"Funny I never noticed this, even though I had to listen to the SSSS until my ears bled during development! But I mostly did use music and the difference there was not that huge.

Anyway, it is a bug. Will be fixed in next release, which should happen very very soon."

New on the Web

Dilwyn Jones has added the Qubide manual to his website:

"A Qubide replacement manual is now available from the QL Documentation page on my Top-Cities website.

http://dilwynjones.topcities.com/qldocs/qldocs.html

Just scroll down to the Qubide Manuals section about three quarters of the way down the page. The zipped file includes copies of the versions 1 manual and supplement for later versions.

For those interested in the sources, I've put

those up on
http://www.dilwyn.uk6.net/qlrom/index.html
It's quite large file (about 340K of download). My grateful thanks to Derek Stewart who sent me the file and explained that Phil Borman and Ron Dunnett had agreed to release it under the GPL (Gnu Public Licence).
There's also a copy of the v2.02 ROM image on the same page."

New on ***Per Witte***'s site:
"Recent addition to Knoware
http://www.witteware.com/knoware
2007/01/26 Addition
MATTOOL - Array manipulation toolkit. New. Includes the infamous QUICKSORT I've been raving about."

To (US)B or Not To BE

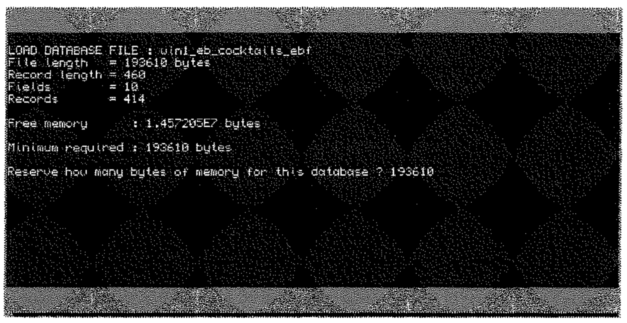
There is a glimmer of hope that the QL could have some form of USB support. QL Today's graphics writer, ***Herb Schaaf***, had spotted an interesting news article:
"Recent issue of Nuts & Volts has article by Jan Axelson. She writes about 'USBwiz' from GHI Electronics. Priced at about $50, allows serial to USB communication. Her site is Lvr.com and ghi is at ghielectronics.com"
Herb asked if this would work with a QL, but opinion among the experts was divided.
Duncan Neithercut was enthusiastic:
"This is the sort of development that should be do able. The device can be driven by I2C commands, the hardware connections are well documented, it is designed for hobbyists, drivers in C are provided, the suppliers claim it can be driven by any processor including Motorola. I2C drivers of a type exist for the QL for the Minerva ROM. no doubt there would be work to do to make it happen but less than designing bespoke hardware from the ground up.
UK suppliers have it on sale for £30 a piece with development rigs for £150 - a lot less than the £1000s suggested before."
Others sounded warnings. ***Tony Firshman*** pointed out that the hardware was the easy bit. We would also need people with the time and expertise to write drivers. ***Marcel Kilgus*** doubted whether the speed would be high enough for devices other than keyboards and mice. Tony added that this might mean it was only usable with Super Hermes. ***Malcolm Cadman*** added that the device had built in drivers for use with keyboards, mice and printers, which prompted a warning from ***Rich Mellor*** that a USB printer would still have to be QL compatible.
Tony Firshman promised to buy a couple of units and test these, and it is probably best not to build up too many hopes at this stage.

Dilwyn Jones Software Update

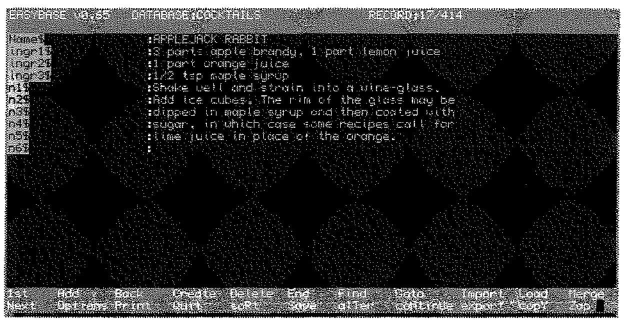
Dilwyn Jones has also updated two of his programs:

EASYBASE

"Easybase version 0.65 has now been released. This contains two bug fixes. The first fixes a bug when running on a Q40 or Q60 where the program tries to switch to mode 0 from high colour modes and finds itself unable to run on high colour modes. The second bug fixed is the count of free memory, where the method of calculation has changed on recent versions of SMSQ/E such as QPC2. This causes a situation where a machine may have, say, 16MB of free memory and Easybase is only able to see less than 1 MB of it.
Easybase is now a freeware program, and it may be downloaded from my website's Databases page:
http://www.dilwyn.uk6.net/database/index.html



```
LOAD DATABASE FILE : win1_eb_cocktails_ebf
File length   = 193610 bytes
Record length = 460
Fields        = 10
Records       = 414

Free memory        : 1.457205E7 bytes

Minimum required : 193610 bytes

Reserve how many bytes of memory for this database ? 193610
```



```
EASYBASE v0.65   DATABASE:COCKTAILS              RECORD:17/414
Name$          :APPLEJACK RABBIT
ingr1$         :3 parts apple brandy, 1 part lemon juice
ingr2$         :1 part orange juice
ingr3$         :1/2 tsp maple syrup
n1$            :Shake well and strain into a wine-glass.
n2$            :Add ice cubes. The rim of the glass may be
n3$            :dipped in maple syrup and then coated with
n4$            :sugar, in which case some recipes call for
n5$            :lime juice in place of the orange.
n6$            :
```

1st Add Back Create Delete End Find Goto Import Load Merge
Next Options Print Quit soRt Save alTer conTinue export "copy" Zap

STIQQIES

The Stiqqies program mentioned in the previous QL Today has now had a small update, giving it a larger input buffer to cope with slightly longer message entries. While Stiqqies was originally only intended to be used for very short reminder messages, some people have asked for it to allow slightly longer messages to be entered, hence this change.

It does not work on version AH or JM QL ROMs though, where the input buffer is strictly limited to 128 characters.
The program is freeware, and the updated version may be downloaded from:
http://www.dilwyn.uk6.net/misc/index.html

Versions
A brief reminder that a list of current version numbers of programs written by me can be found at:
http://www.dilwyn.uk6.net/versions/ index.html
This list is available in HTML, Abacus spreadsheet or plain text versions."

Down Memory Lane

Rick Chagouri-Brindle recently posted an interesting email on the QL-users list:
"I just thought I'd share a recent QL experience with everybody! The company I work for is very keen on promoting personal development outside the normal boundaries at work. Think outside the box is one of the MD's catch-phrases.
Anyway, as IT Manager of this company, it became my turn to organise something concerning my field but not directly work related: so I organised a little presentation and history tour, using my collection of PCs as illustrations. People were able to have a go, play with them, run programs and generally see how computers have developed over the years. To illustrate my presentation, I used my ZX81 (the first PC I ever owned), BBC Model B, BBC Master, Spectrum+3, Amiga 1200, Z88 and of course the Sinclair QL.
It was wonderful to see the amount of fun people had with these older machines, and some of the younger members of the company were really surprised at what had constituted a computer "back then"! It's such as shame that for many youngsters today, learning ICT (as the schools irritatingly insist on calling it) is simply learning how to use Microsoft products . . . .
One of the most interesting comments was how "cool" the QL looked . . . it seems that well-designed retro is in!!!!"
Those readers who were at QL2004 will remember the talk on QL history that **Urs König** gave. He replied to Rick's email:
"One of my remaining QL tasks is to let the ITC profs remember from where the real experts/technologies are (were) coming from. Doing that I experienced the same impressions as Rick did. Once the young ITC profs hear and see what and how a QL and QDOS operated some 20 years ago they get very interested/ impressed.
Eg. Our company is in a network of about 30 ITC companies. In summer 2004 the monthly "First monday" event was organised by me around my private QL thing. It was named "The roots, the cousins, the innovations". We had a very special guest. Tim Bucher, cousin of my wife and Vice President at Apple (at that time). Both Tim and I had very personal speeches and presentations. I did some live QL demos. Eg. having › 100 jobs running in parallel under SMSQ/E compared to Windows NT 4 where NT went down on that. Tim did impress us all with his inventions. My computer museum was on display with different QL systems running software. We even had a working original Macintosh from 1984.
The 40 or so attendees were impressed and even today, almost 3 years later people talk about it one day or another."
He also gave some links:
http://www.computervalley.ch/website/veranstaltungen/firstmonday/020804_TimBucher.htm
http://www.computervalley.ch/website/veranstaltungen/firstmonday/Foto_Gallery_Tim_Bucher.htm
http://www.computervalley.ch/020804/Besuch_Tim_Bucher.pdf
http://mypage.bluewin.ch/QLvsJaguar/QL.html

Turbo History

Recently on the QL-users email list, there was a request for more practice information on Turbo and the pointer environment. These are the articles that have been published in QL Today:
V5i1 page 11 - Turbo and the Pointer Environment - George Gwilt
V5i4 page 7 - The New Turbo Compiler - Dilwyn Jones and George Gwilt
V5i5 page 25 - TurboPTR and Sprites/Blobs/ Patterns on the Q40 - George Gwilt
V6i1 page 50 - Turbo version 4.8 - George Gwilt
V6i5 page 21 - TurboPTR and QMenu - George Gwilt
V7i4 page 39 - "Out of Range" or PE Windows Tamed - George Gwilt
V7i5 page18 - "PE Windows - the Orthodox Way" - George Gwilt
V8i3 page 20 - "SMSQ/E v3.xx and PE" - George Gwilt
V8i4 page 49 - "QL Extras" - George Gwilt
V8i5 page 11 - "Turbo and Parameters" - George Gwilt
V8i5 page 34 - "Windows in the Pointer Environment using SETW" - George Gwilt
V8i5 page 37 - "Tiptoeing Through Turbo" -

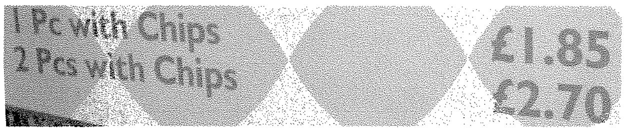
George Gwilt
V8i6 page 16 - "EasyPtr and TurboPtr" - George Gwilt
V8i6 page 27 - "Turbo Revisited" - David Denham
V10i1 page 39 - "Turbo v.4.21" - George Gwilt
V11i1 page 50 - "Turbo and Config Blocks" - George Gwilt

Head Hung Low

QL Today's editor hangs his head in shame. Having roundly castigated the Quanta committee in recent editions for not knowing their own constitution, he made his own blunder in the last QL Today. In his report of the Byfleet show he wrote that there was "no minimum quorum" for Quanta AGMs. This is incorrect. Article 8.0 of the constitution states that the AGM quorum is 10.
In his defence your editor says that at least he admits to his constitutional sins even when no one else has spotted them, which is more than can be said of the Quanta committee.
And just a final thought. Only 14 members attended the 2006 Quanta AGM. It was a close shave.

PC's going cheap?

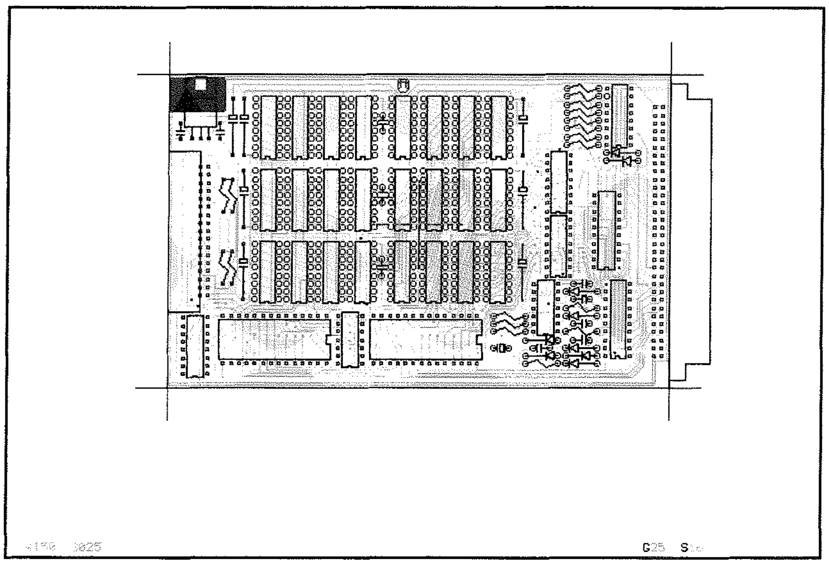
During his occasional forays south of the Thames for nefarious purposes your editor came across what would seem to be the computer bargain of the century.
Before QL Today readers travel en masse to South London to snap up this offer, we should perhaps warn you that on closer examination these PC's are pieces of chicken. More going cluck, cluck than cheap.



LEAR PCB CAD Program Update

Malcolm Lear has released version 6.12 of his PCB CAD program. This is available for download from Dilwyn Jones's website, at:
http://www.dilwyn.uk6.net/graphics/index.html
There are lots of changes in this version, not the least of which is the massive library in this package, larger than even some PC programs such as CIRCAD. Which unfortunately means that the zip file which is on the website is now approximately twice the size of the previous version on the page.
Here is a list of changes to recent versions of this program:

6.05 Gerber output bug removed.
6.06 Up/Down directory navigation added to file load system.
6.07 Library and Art files have separate colour schemes.
Press 'D' for default colour schemes added.
File names can now be lower case and 1 character longer.
Track width for 0 and 1 can be set to 5th and 7.5th
Info blocks can be increased.
More user parameters are save in the file. Including track,pad, text sizes and layers.
6.08 Sorted change scale bug (hopefully).
Extra large scale view modes for high resolution work.
Library files are now designated by the .lib extension.
Huge supply of Jedec standard SMD library packages.
'Cntrl/Shift S' saves library .lib files.
Cadtk.ext now compiled using GWASS rather than GST
6.09 Block elements are extended to have multiple segments to allow the creation of many types of SM devices.
6.10 Cleaned up gerber output. Gerber output now has options for leading zero suppression and modal coordinates.
6.11 Automatic layer re assignment from 2 to 11 and 6 to 10.
Libraries layers reassigned.
Null length gerber files cleaned up and deleted.
6.12 Fast single key gerber generation.



JOCHEN MERZ SOFTWARE

Kaiser-Wilh.-Str. 302 D-47169 Duisburg
Tel. 0203 502011 Fax 0203 502012
http://smsq.j-m-s.com smsq@j-m-s.com

QDT - The QL DeskTop

QDT brings the QL a modern, easy- and intuitive-to-use graphical user interface. Arrange your programs, games, applications in folders, start your favourite applications with a single click. Fully configurable!
QDT comes with an automatic installer – like a guided tour. Installing it is as easy as never before!

System requirements:
- SMSQ/E Version 3.06 or later
- Graphics Driver 2 Colour support
- HARD DRIVE with at least 3MB free space.
- 4MB RAM

What do you get when you order QDT?
- A floppy disk for Aurora or quick-start users.
- A CD ROM containing a QXL.win file, backgrounds, sprite editor, and many other extras.
- A printed quick-start guide

QDT costs EUR 59,90 plus postage (EUR 3,– Europe, EUR 6,– rest of the world).

BALLED! - game for the Pointer Environment

Balled is the new Game from Wolfgang Lenerz for QDOS and SMSQ/E. It runs in high-colour mode (looks great) and can also be played in standard QL Mode 4.
Also, Balled uses the digital sound system if you run it on a recent version of SMSQ/E and QPC, Q40 and Q60! The aim of the game is to build lines by moving coloured balls around, following some rules, of course. After every move, more balls appear, so you have to be careful not to fill up the board – then the game is over! Different levels of difficulty and wildcard balls are also featured. Find out how to get bonus and higher scores to fill up the high-score table!
Balled costs only EUR 11,90 plus postage (EUR 2,- Europe, EUR 4,- rest of the world).

QPCPrint - printer driver for QPC

QPCPrint allows you to print to virtually any printer connected to your PC running QPC, even to fax and pdf printer drivers. It accepts EPSON ESC/P2 codes to any PAR printer and converts it to output which can be handled by Windows, which looks very similar to original EPSON output. You can even configure the individual fonts used by the emulation. This is THE application many QPC users waited for a long time!
QPCPrint costs only EUR 39,90 plus postage (EUR 3,- worldwide - shipped on CD)
Free Delivery via email now possible: http://Printer-Emulator.J-M-S.com

ZX8301 .... only EUR 4,- each (plus 2,50 EUR postage for up to 5 ICs)
ZX8302 .... only EUR 2,- each (plus 2,50 EUR postage for up to 5 ICs)

Still searching for **DD Disks**? Well, I've got them as well. New, in boxes of 10 ... various colours or just blue: Pack of 10 DD disks for only EUR 4,- each (plus 4,50 EUR postage for 1 or 2 packs, 9 EUR for up to 7 packs)

If you wish to place your order via internet, please do not send your credit card details in an email!
Please use the secure contact form on the SMSQ homepage: SMSQ.J-M-S.COM
We accept VISA, MasterCard and Diners Club!

Enigma

by Ian Pine

This program is based on the Enigma Machine used during World War II to encipher messages for military purposes. My knowledge of the machine is limited to the brief description contained in a booklet I bought when I visited Bletchley Park - where the main wartime efforts to break the cipher were made - but all the components described have been implemented, though some of the internal 'wiring' has been subject to 'educated' guesswork on my part where the booklet did not go into enough detail.

The two most fundamental features of the original system have been retained:

(a) it is a 'self-inverting' cipher, i.e. the original plain-text can be recovered simply by typing the cipher-text back into a machine in the same initial state;

(b) the plain-text letter and the cipher-letter can never be the same.

The main components of the machine are:

(1) a keyboard comprising the 26 alphabetic characters, (2) a plug-board, (3) a set of three rotors, (4) a 'reflector' and (5) a set of 26 lamps in the same layout as the keyboard.

The plug-board allows pairs of letters to be crossed over before being applied to the rotor system. This means that if, for example, A and D are crossed over and the A key is pressed then D will be sent to the rotors. It also means that if A is returned from the rotors then the D lamp will be lit. Protocol required ten pairs of letters to be selected.

The three rotors are linked in the style of a mechanical counter, i.e. for each complete revolution of one rotor, the one to its left will be moved on one position. As the rotors revolve, one step for each keypress, the hard-wired electrical path through them realigns thus changing the cipher letter returned even if the same letter key is repeatedly pressed. The position on the middle and righthand rotors where the turn-over is triggered is user selectable and known as the "ring-setting". Once selected it remains fixed for the entire message.

The reflector simply loops the electrical path emerging from the lefthand rotor back through all three in the reverse order via a different route.

The electric current re-emerging from the righthand rotor is fed back through the plug-board, to illuminate one of the 26 lamps.

For a much better explanation I recommend a visit to Bletchley Park or the website of the Bletchley Park Trust: **www.bletchleypark.org.uk**, where you will find, among a huge amount of content, a downloadable Windows version of the Enigma simulator.

A few statistics (from the booklet): There are 10 ways of selecting 3 rotors from the set of 5; there are 6 orders for placing them in the machine; 26x26x26 (17576) start positions for the rotors; and there are about 1.5x10^14 ways of selecting 10 pairs of letters on the plug-board. In all, you can set your machine up at the start of a session in about 1.58x10^20 (158 million, million, million) ways.

The main menu of the program is colour coded: items in red are not yet available; items in white may be selected; the initial settings items (1..4) will change to green when they have been completed. Brief instructions on each screen are given so you know what options are available. You might just want to reset the rotors back to their start positions to start a new message with the same settings; the program allows this without having to reset all the settings and start again.

Notes: The Restore Configuration option does not validate the contents of the file given; the program will either break or fail to operate correctly if bad data is supplied. The Save Configuration will break (in QDOS) if the file already exists. If the program breaks on opening the file, try:

```
LET f$="<correct filename>":RETRY
```

I have provided a couple of short programs to make a new set of five rotors and a new reflector. Run the programs then MERGE the data files into the main program - beware line number changes if you modify the main program! If the `LANGUAGE` keyword is not available, replace it with the international code for your country. I have provided keyboard layouts for French, English and German; if you think of any different ones simply add them to the list in the DATA section and modify line 190. Keep the 0,"","","" delimiter line!

Test example:

Plug board: `AC BM DJ EN FP GR HL IT KO SW`

Rotors: 3 (left), 1 (middle), 4 (right)

Rings:	-	B	K
Start:	B	N	S

```
XMGCW PLHZT CEHLH IZCNL UIIIK QRYAL YORPZ
```

Main program Enigma

```
100 REMark (c) 2006 Ian Pine.
    QL Enigma Machine.  enigma_bas.  v1.0
110 DIM pb%(25),ro%(4,25),ri%(2,25),rr%(2,25),
    rf%(25),rs%(2),gr%(4)
120 DIM sp%(2),cp%(2),cx(25),cy(25),kcx%(25),
    kcy%(25),kr$(2,10)
130 WINDOW 512,256,0,0
140 PAPER 0
150 CLS
160 :
170 REMark Establish keyboard layout.
180 LET l=LANGUAGE
190 IF l<>33 AND l<>44 AND l<>49 THEN LET l=44
200 RESTORE
210 REPeat GetLang
220  READ l1,r1$,r2$,r3$
230  IF l1=0 THEN EXIT GetLang
240  IF l1=l THEN
250   LET kr$(0)=r1$
260   LET kr$(1)=r2$
270   LET kr$(2)=r3$
280  END IF
290 END REPeat GetLang
300 FOR i=0 TO 2
310  FOR j=0 TO LEN(kr$(i))-1
320   LET p=CODE(kr$(i,j+1))-65
330   LET kcx%(p)=j*10+30+i*2
340   LET kcy%(p)=27-i*10
350  END FOR j
360 END FOR i
370 :
380 REMark Initialize settings.
390 ResetPlugBoard
400 ResetRotorStart
410 ResetRingSetting
420 LET rodone=0
430 LET pbdone=0
440 REMark Pre-calculate coordinates for
    printing rotor alphabet in a circle.
450 FOR i=0 TO 25
460  LET cx(i)=20*SIN(PI*i/13)
470  LET cy(i)=20*COS(PI*i/13)
480 END FOR i
490 REMark Load wiring definitions for rotor
    set.
500 FOR i=0 TO 4
510  FOR j=0 TO 25
520   READ ro%(i,j)
530  END FOR j
540 END FOR i
550 REMark Load reflector 'wiring'.
560 FOR i=0 TO 25
570  READ rf%(i)
580 END FOR i
590 :
600 DEFine PROCedure ResetPlugBoard
610 LOCal i
620  FOR i=0 TO 25
630   LET pb%(i)=i
640  END FOR i
650  LET pbc=0
660  LET pbdone=0
670 END DEFine ResetPlugBoard
680 :
690 DEFine PROCedure ResetRotorStart
700  LET sp%(0)=0
710  LET sp%(1)=0
720  LET sp%(2)=0
730  LET cp%(0)=0
740  LET cp%(1)=0
750  LET cp%(2)=0
760  LET spdone=0
770 END DEFine ResetRotorStart
780 :
790 DEFine PROCedure ResetRingSetting
800  LET rs%(1)=0
810  LET rs%(2)=0
820  LET rsdone=0
830 END DEFine ResetRingSetting
840 :
850 REPeat MainMenu
860  LET r=0
870  WINDOW #0,512,52,0,204
880  PAPER #0,0
890  INK #0,4
900  BORDER #0,1,255
910  CSIZE #0,0,0
920  CLS #0
930  WINDOW 512,202,0,0
940  PAPER 0
950  INK 6
960  BORDER 1,255
970  CSIZE 0,0
980  CLS
990  PRINT "Main Menu"
1000  IF pbdone THEN INK 4
1010  PRINT \"1. Plug-board setup";
1020  INK 6
1030  IF rodone THEN INK 4
1040  PRINT \"2. Rotor selection";
1050  INK 6
1060  IF rodone=0 THEN INK 2
1070  IF rsdone THEN INK 4
1080  PRINT \"3. Ring settings";
1090  INK 6
1100  IF rodone=0 THEN INK 2
1110  IF spdone THEN INK 4
1120  PRINT \"4. Rotor start positions";
1130  INK 6
1140  PRINT \"5. Clear all settings and start
      again"
1150  PRINT "6. Restore a saved configuration"
1160  IF NOT(AllDone) THEN INK 2
1170  PRINT "7. Save current configuration"
1180  INK 6
1190  IF NOT(AllDone) THEN INK 2
1200  PRINT "8. Start enciphering/deciphering
      a message"
1210  INK 6
1220  PRINT "9. Exit"
1230  INPUT #0,"Enter selection number
      (1..9)"\r$
1240  IF r$<>"" THEN IF r$(1)>="0" AND
      r$(1)<="9" THEN LET r=r$
1250  IF r=0 THEN
1260   PRINT #0,"Selection not valid; try
       again."
1270  ELSE
1280   SELect ON r
1290    =1
1300     PlugBoard
1310    =2
1320     SelectRotors
1330    =3
1340     IF rodone THEN RingSettings
1350    =4
1360     IF rodone THEN RotorStart
1370    =5
1380     ResetPlugBoard
1390     LET rodone=0
1400     ResetRotorStart
```

```
1410     ResetRingSetting
1420    =6
1430     RestoreConfig
1440    =7
1450     IF AllDone THEN SaveConfig
1460    =8
1470     IF AllDone THEN Cipher
1480    =9
1490     WINDOW 256,202,256,0
1500     PAPER 0
1510     INK 6
1520     BORDER 1,255
1530     CSIZE 0,0
1540     CLS
1550     BORDER #2,1,255
1560     CLS #2
1570     PRINT #0,"Program ends."
1580     STOP
1590   END SELect
1600  END IF
1610 END REPeat MainMenu
1620 :
1630 DEFine FuNction AllDone
1640  RETurn pbdone AND rodone AND spdone AND
      rsdone
1650 END DEFine AllDone
1660 :
1670 DEFine PROCedure ShowStecker
1680 LOCal i
1690  CLS
1700  FOR i=0 TO 25
1710   CURSOR i*5+10,45,0,0
1720   PRINT CHR$(65+i)
1730   CURSOR i*5+10,75,0,0
1740   PRINT CHR$(65+i)
1750   LINE i*5+11,46 TO pb%(i)*5+11,70
1760   IF pb%(i)>i THEN
1770    INK 2
1780    CURSOR i*5+10,5,0,0
1790    PRINT CHR$(65+pb%(i))
1800    CURSOR i*5+10,35,0,0
1810    PRINT CHR$(65+i)
1820    LINE i*5+11,6 TO i*5+11,30
1830    INK 6
1840   END IF
1850  END FOR i
1860 END DEFine ShowStecker
1870 :
1880 DEFine PROCedure PlugBoard
1890 LOCal k1,k2
1900  CLS
1910  REPeat PBloop1
1920   CLS #0
1930   PRINT #0,"Enter two [different] letters
       to set, or letter followed by * to
       unset."
1940   PRINT #0,"Repeat first letter to cancel
       it."
1950   REPeat PBloop2
1960    ShowStecker
1970    AT 0,3
1980    PRINT "Select ten plug-board pairings"
1990    AT 2,3
2000    CSIZE 2,1
2010    PRINT pbc
2020    CSIZE 0,0
2030    IF pbc=10 THEN EXIT PBloop2
2040    LET k1=GetLetter(0)
2050    CURSOR k1*5+10,75,0,0
2060    INK 2
2070    PRINT CHR$(k1+65)
2080    INK 6
2090    LET k2=GetLetter(1)
2100    IF k2=42 THEN
2110     IF pb%(k1)<>k1 THEN
2120      LET k2=pb%(k1)
2130      LET pb%(k1)=k1
2140      LET pb%(k2)=k2
2150      LET pbc=pbc-1
2160     END IF
2170    ELSE
2180     IF k2<>k1 AND pb%(k1)=k1 AND
         pb%(k2)=k2 THEN
2190      LET pb%(k1)=k2
2200      LET pb%(k2)=k1
2210      LET pbc=pbc+1
2220     END IF
2230    END IF
2240   END REPeat PBloop2
2250   IF Confirm("Confirm plug-board
       selections are correct") THEN EXIT
       PBloop1
2260   ResetPlugBoard
2270  END REPeat PBloop1
2280  LET pbdone=1
2290 END DEFine PlugBoard
2300 :
2310 DEFine PROCedure SelectRotors
2320 LOCal i,k,n
2330  REPeat SRloop1
2340   CLS
2350   CSIZE 2,1
2360   FOR i=0 TO 4
2370    LET gr%(i)=-1
2380    CIRCLE 20+i*32,20,15
2390    CURSOR 20+i*32,20,-6,-10
2400    PRINT i+1
2410   END FOR i
2420   CSIZE 0,0
2430   CLS #0
2440   PRINT #0,"Select three rotors in the
       sequence right, middle, left."
2450   PRINT #0,"Type the number of the rotor
       [1..5]."
2460   FOR n=2 TO 0 STEP -1
2470    CURSOR 30+45*n,70,-3,-5
2480    PRINT "?"
2490    REPeat SRloop2
2500     LET k=GetRotor
2510     IF gr%(k)=-1 THEN EXIT SRloop2
2520    END REPeat SRloop2
2530    CopyRotor n,k
2540    INK 0
2550    FILL 1
2560    CIRCLE 20+k*32,20,15
2570    FILL 0
2580    LET gr%(k)=n
2590    INK 6
2600    CURSOR 30+45*n,70,-3,-5
2610    PRINT k+1
2620    DrawRotor n
2630   END FOR n
2640   IF Confirm("Confirm rotor selection is
       correct") THEN EXIT SRloop1
2650  END REPeat SRloop1
2660  LET rodone=1
2670 END DEFine SelectRotors
2680 :
2690 DEFine PROCedure CopyRotor(n,k)
2700 LOCal i
2710  FOR i=0 TO 25
2720   LET ri%(n,i)=ro%(k,i)
2730   LET rr%(n,ro%(k,i))=i
2740  END FOR i
```

```
2750  LET sp%(n)=0
2760  LET cp%(n)=0
2770  LET rs%(n)=0
2780 END DEFine CopyRotor
2790 :
2800 DEFine PROCedure DrawRotor(n)
2810 LOCal i
2820  FOR i=0 TO 25
2830   INK 6
2840   CURSOR 30+45*n+cx(i),70+cy(i),-3,-5
2850   IF n>0 AND rs%(n)=(cp%(n)+i) MOD 26
       THEN INK 2
2860   PRINT CHR$(65+(cp%(n)+i) MOD 26)
2870  END FOR i
2880  INK 6
2890  CURSOR 30+45*n,95,-3,-5
2900  PRINT CHR$(191)
2910 END DEFine DrawRotor
2920 :
2930 DEFine FuNction GetRotor
2940 LOCal k
2950  REPeat GRloop1
2960   LET k=CODE(INKEY$(-1))
2970   IF k<49 OR k>53 THEN NEXT GRloop1
2980   RETurn k-49
2990  END REPeat GRloop1
3000 END DEFine GetRotor
3010 :
3020 DEFine FuNction GetLetter(f)
3030 LOCal k
3040  REPeat GLloop1
3050   LET k=CODE(INKEY$(-1))
3060   IF f=1 AND k=42 THEN RETurn k
3070   IF k>96 AND k<123 THEN LET k=k-32
3080   IF k<65 OR k>90 THEN NEXT GLloop1
3090   RETurn k-65
3100  END REPeat GLloop1
3110 END DEFine GetLetter
3120 :
3130 DEFine FuNction Confirm(m$)
3140 LOCal r$
3150  CLS #0
3160  REPeat Cloop1
3170   INPUT #0,m$&" (y/n):"\r$
3180   IF r$=="y" OR r$=="yes" THEN RETurn 1
3190   IF r$=="n" OR r$=="no" THEN RETurn 0
3200   PRINT #0,"Please answer y[es] or n[o]."
3210  END REPeat Cloop1
3220 END DEFine Confirm
3230 :
3240 DEFine PROCedure RingSettings
3250 LOCal k,n
3260  REPeat RSloop1
3270   CLS
3280   FOR n=0 TO 2
3290    DrawRotor n
3300   END FOR n
3310   FOR n=2 TO 1 STEP -1
3320    CLS #0
3330    PRINT #0,"Select ring setting for the
        ";
3340    IF n=2 THEN PRINT #0,"righthand";
3350    IF n=1 THEN PRINT #0,"middle";
3360    PRINT #0," rotor [A..Z]:"
3370    LET k=GetLetter(0)
3380    LET rs%(n)=k
3390    DrawRotor n
3400   END FOR n
3410   IF Confirm("Confirm ring-settings are
       correct") THEN EXIT RSloop1
3420  END REPeat RSloop1
3430  LET rsdone=1
3440 END DEFine RingSettings
3450 :
3460 DEFine PROCedure RotorStart
3470 LOCal k,n,tp%(2)
3480  REPeat SPloop1
3490   CLS
3500   CLS #0
3510   PRINT #0,"Enter * to return to last set
       start positions."
3520   FOR n=0 TO 2
3530    DrawRotor n
3540   END FOR n
3550   FOR n=2 TO 0 STEP -1
3560    AT #0,1,0
3570    PRINT #0,"Select start position for
        the ";
3580    IF n=2 THEN PRINT #0,"righthand";
3590    IF n=1 THEN PRINT #0,"middle";
3600    IF n=0 THEN PRINT #0,"lefthand";
3610    PRINT #0," rotor [A..Z]:   "
3620    LET k=GetLetter(1)
3630    IF k=42 THEN
3640     FOR k=2 TO 0 STEP -1
3650      REPeat SPloop3
3660       IF cp%(k)=sp%(k) THEN EXIT SPloop3
3670       AdvanceRotor k
3680       PAUSE 6
3690      END REPeat SPloop3
3700      LET tp%(k)=sp%(k)
3710     END FOR k
3720     EXIT n
3730    END IF
3740    LET tp%(n)=k
3750    REPeat SPloop2
3760     IF cp%(n)=k THEN EXIT SPloop2
3770     AdvanceRotor n
3780     PAUSE 6
3790    END REPeat SPloop2
3800   END FOR n
3810   IF Confirm("Confirm start positions are
       correct") THEN EXIT SPloop1
3820  END REPeat SPloop1
3830  LET sp%(0)=tp%(0)
3840  LET sp%(1)=tp%(1)
3850  LET sp%(2)=tp%(2)
3860  LET spdone=1
3870 END DEFine RotorStart
3880 :
3890 DEFine PROCedure AdvanceRotor(n)
3900  IF n<0 OR n>2 THEN RETurn
3910  LET cp%(n)=(cp%(n)+1) MOD 26
3920  DrawRotor n
3930  IF n>0 THEN
3940   IF rs%(n)=(cp%(n)-1) MOD 26 THEN
       AdvanceRotor n-1
3950  END IF
3960 END DEFine AdvanceRotor
3970 :
3980 DEFine PROCedure Cipher
3990  LOCal lk
4000  INK 6
4010  PAPER 0
4020  CLS
4030  FOR lk=0 TO 2
4040   DrawRotor lk
4050  END FOR lk
4060  FOR lk=0 TO 25
4070   CIRCLE kcx%(lk),kcy%(lk),4
4080   CURSOR kcx%(lk),kcy%(lk),-3,-5
4090   PRINT CHR$(lk+65)
4100  END FOR lk
4110  CLS #0
```

```
4120  PRINT #0,"Enter the letters of your
      message.  The corresponding cipher
      letter is"
4130  PRINT #0,"indicated by the illuminated
      'lamp'.  Finish by typing *."
4140  LET lk=0
4150  REPeat CipherLoop
4160   LET nk=GetLetter(1)
4170   IF nk=42 THEN EXIT CipherLoop
4180   INK 0
4190   FILL 1
4200   CIRCLE kcx%(lk),kcy%(lk),4
4210   FILL 0
4220   INK 6
4230   CIRCLE kcx%(lk),kcy%(lk),4
4240   CURSOR kcx%(lk),kcy%(lk),-3,-5
4250   PRINT CHR$(lk+65)
4260   LET lk=Encode(nk)
4270   INK 4
4280   FILL 1
4290   CIRCLE kcx%(lk),kcy%(lk),4
4300   FILL 0
4310   PAPER 4
4320   INK 0
4330   CURSOR kcx%(lk),kcy%(lk),-3,-5
4340   PRINT CHR$(lk+65)
4350   PAPER 0
4360   AdvanceRotor 2
4370  END REPeat CipherLoop
4380 END DEFine Cipher
4390 :
4400 DEFine FuNction Encode(k)
4410 LOCal i,o
4420  LET o=pb%(k)
4430  FOR i=2 TO 0 STEP -1
4440   LET o=(ri%(i,(cp%(i)+o) MOD 26)-cp%(i))
       MOD 26
4450  END FOR i
4460  LET o=rf%(o)
4470  FOR i=0 TO 2
4480   LET o=(rr%(i,(cp%(i)+o) MOD 26)-cp%(i))
       MOD 26
4490  END FOR i
4500  RETurn pb%(o)
4510 END DEFine Encode
4520 :
4530 DEFine PROCedure SaveConfig
4540 LOCal i
4550  CLS #0
4560  INPUT #0,"Enter the name of the file you
      want to save settings to:"\f$
4570  OPEN_NEW #3,f$
4580  FOR i=0 TO 25
4590   PRINT #3,pb%(i)
4600  END FOR i
4610  PRINT #3,gr%(0)\gr%(1)\gr%(2)\gr%(3)\gr%(4)
4620  PRINT #3,sp%(0)\sp%(1)\sp%(2)
4630  PRINT #3,rs%(1)\rs%(2)
4640  CLOSE #3
4650 END DEFine SaveConfig
4660 :
4670 DEFine PROCedure RestoreConfig
4680 LOCal i
4690  CLS #0
4700  INPUT #0,"Enter the name of the file
      containing your settings:"\f$
4710  OPEN_IN #3,f$
4720  FOR i=0 TO 25
4730   INPUT #3,pb%(i)
4740  END FOR i
4750  LET pbdone=1
4760  LET pbc=10
4770  FOR i=0 TO 4
4780   INPUT #3,gr%(i)
4790   IF gr%(i)<>-1 THEN CopyRotor gr%(i),i
4800  END FOR i
4810  LET rodone=1
4820  INPUT #3,sp%(0),sp%(1),sp%(2)
4830  LET cp%(0)=sp%(0)
4840  LET cp%(1)=sp%(1)
4850  LET cp%(2)=sp%(2)
4860  LET spdone=1
4870  INPUT #3,rs%(1),rs%(2)
4880  LET rsdone=1
4890  CLOSE #3
4900 END DEFine RestoreConfig
4910 :
4920 REMark Localized keyboard layouts
4930 DATA 33,"AZERTYUIOP","QSDFGHJKLM","WXCVBN"
4940 DATA 44,"QWERTYUIOP","ASDFGHJKL","ZXCVBNM"
4950 DATA 49,"QWERTZUIOP","ASDFGHJKL","YXCVBNM"
4960 DATA 0,"","",""
4970 REMark 'Wiring' definitions for the five
     rotors
4980 DATA 13,6,1,8,19,2,14,16,9,10,3,11,4,0,22
     ,20,5,7,23,15,21,12,18,25,24,17
4990 DATA 19,23,17,13,5,11,15,16,7,0,14,3,12,2
     ,21,22,1,6,9,10,4,8,20,18,25,24
5000 DATA 18,9,0,14,22,11,8,10,12,15,13,19,25,
     1,7,3,2,23,17,20,21,16,6,4,5,24
5010 DATA 10,1,5,2,23,3,13,6,8,25,20,19,21,11,
     7,0,14,12,9,16,18,22,17,24,15,4
5020 DATA 12,22,13,7,15,2,25,17,3,1,14,18,0,4,
     5,11,6,19,23,20,21,24,9,16,8,10
5030 REMark 'Wiring' definition for the
     reflector
5040 DATA 6,24,16,19,9,7,0,5,15,4,18,25,17,23,
     21,8,2,12,10,3,22,14,20,13,1,11
```

Make a new Reflector

Load the main program then MERGE the data file.

```
110 OPEN_NEW #3,win3_enigma_reflector_dat
120 DIM r%(25)
130 RANDOMISE
140 FOR i=0 TO 25
150  LET r%(i)=i
160 END FOR i
170 FOR i=0 TO 25
180  IF r%(i)=i THEN
190   REPeat lp1
200    LET a%=RND(25)
210    IF a%<>i AND r%(a%)=a% THEN EXIT lp1
220   END REPeat lp1
230   LET r%(i)=a%
240   LET r%(a%)=i
250  END IF
260 END FOR i
270 PRINT #3,"5040 DATA ";
280 FOR i=0 TO 24
290  PRINT #3,r%(i);",";
300 END FOR i
310 PRINT #3,r%(25)
320 CLOSE #3
```

Make five new rotors

```
Load the main program then MERGE the data file.
110 OPEN_NEW #3,win3_enigma_rotors_dat
120 RANDOMISE
130 DIM r%(25)
140 LET lnum=4980
150 FOR rotor=1 TO 5
160  FOR i=0 TO 25
170   LET r%(i)=i
180  END FOR i
190  PRINT #3,lnum;" DATA ";
200  LET last=25
210  REPeat rlp1
220   IF last=0 THEN EXIT rlp1
230   LET a=RND(last-1)
240   PRINT #3,r%(a);",";
250   LET r%(a)=r%(last)
260   LET last=last-1
270  END REPeat rlp1
280  PRINT #3,r%(0)
290  LET lnum=lnum+10
300 END FOR rotor
310 CLOSE #3
```

Using QPC under Linux: an example

by Wolfgang Lenerz

I'm in the (slow) process of switching to Linux, since I just don't think that Windows Vista is for me (I'm still using win98...).
The main reason I've kept off Linux until now was that I couldn't use QPC which, as we all know, runs under Windows. Quite some time ago Marcel pointed me to a software called "wine" which should be able to run QPC under Linux.
Wine is one of those ridiculous acronyms of which the Linux world is so fond (Wine Is Not an Emulator) and it basically allows some Windows programs to run under Linux by pretending to these programs that they are running under Windows. If you're interested in what it can do or not you should have a look at **www.winehq.com**.
Of course, I started off by downloading and installing some Linux variant, Suse Linux 10.2 in my case. I won't go here into how you download and install that, suffice it to say that it was a pretty painless experience - download the CD (or DVD) image(s), burn them to discs and feed the disc(s) to the computer.
After I'd tinkered with the resulting Linux for a few weeks, I thought I'd have a go at installing QPC under it. First I downloaded the wine "rpm" file for my Suse version from **www.winehq.com** (follow the links there). The file was a so-called "rpm" file which is a special "package" file and it was called "**wine-0.9.29-SuSElinux102.i586.rpm**". Please note that these *-/8*"# Unix derivates like Linux treat lower and upper case differently (whoever thought that up deserves to be shot - and, oh, am I gonna get flak for that remark), so a file called "**wine-0.9.29-SuSElinux102.i586.rpm**" is not the same as a file called "**wine-0.9.29-SUSElinux102.i586.rpm**" (can you spot the difference?).
This was downloaded into the download directory and then I copied it into another directory, called "tmp" in my case. I used the graphical file manager called "Konqueror" for this, which is the equivalent of the QPAC2 Files menu, or the Windows Exploder.
Now this package file had to be unpacked. However, to unpack fhis file, you need to be a Superuser, generally called "root". So I logged out of the "session" I was using for the download and logged in as root. I presume that you know how to do that. (I was later told that I could also have done this directly from a normal user terminal window using the command "su").
The unpacking, apparently, is best done from a "terminal", what we QLers call a a command line. So, I opened a terminal window and went to the directory where the rpm file was stored, i.e., in my case, the `"tmp"` directory: In the terminal window I typed `"cd /tmp"` (notice the space). I then, still as a root user, unpacked this file by typing : `"rpm -Uvh wine-0.9.29-SuSElinux102.i586.rpm"` - oh yes, you have to type the entire name, and remember, don't confuser upper and lower case - if you write the "SuSE" in the name above "SUSE", it won't work (I was later told that you don't have to type the entire name at all, using the TAB key, apparently Linux will fill in the rest of the name once you have typed in a few characters of the beginning of that name).
Once I had done this, the computer went chugging away. Once it was done, I didn't really know about it because it didn't tell me, I just waited for a few minutes... Anyway, I checked and it had installed wine all by itself in a directory called **/usr/lib/wine**.
Now I logged off as root and logged on as mere mortal again. I opened a terminal window and typed `"wine"`. This brought back an error since wine expects to be given the name of a program to be started. However, by calling it once, wine installed some files in my /home directory (which, for the user I was logged in as, was called **/home/wolf**). So I looked in my home directory

with a file manager to see what had happended there.
If you tell your file manager to display all files including the hidden files (they don't do that by default), you will see a directory called "**.wine**" in the home directory (apparently, the dot at the beginning means that the file is a hidden file).
I opened the "**.wine**" directory - there are several files and directories (folders) in there. The files seem to be emulating the windows registry and the directories were called "**drive_c**" and "**dosdevices**". "**drive_c**" is a folder (directory) that makes programs running under wine believe that this is the Windows "**C**" drive. The "**dosdevices**" directory contains "links" to what the programs running under wine will believe are the different hard disks under windows. Thus a link called "**c:**" points to the "**drive_c**" directory mentioned above and this will then be the C drives for programs under windows. You could, for example, copy your QPC files and the qxl.win file into that folder, and you'd have qpc on drive "**C**" under "**wine**".
However, that's not (yet) what I wanted: I haven't given up on windows altogether and will continue using it for some time, so I wanted the "wine" QPC to be the one I normally use, and the same for the QXL.WIN file. That would not be possible if the qxl.win file was in a folder called "drive_c" on my linux partition, since Windows, of course, can't access Linux partitions, so the qxl.win file would effectivelly be lost to Windows. My QXL.WIN file and QPC itself were on my "true" windows C drive, not the C drive wine thinks it is (the "**drive_c**" folder). You can delete the link that points to the "**drive_c**" folder and make a new link to the true windows C drive (making these links is pretty easy from within Konqueror - drag the true C drive to the "**dosdevices**" folder and drop it there, you will be asked whether you want to copy it there or to create a link to it there, which is what I did at first). So I did try that, but experienced some crashes when running the "wine" QPC. This is probably due to the fact that my true Windows partition is NOT on the C drive but on the E drive, and probably "wine" expects some particular file on the C drive (as you can see if you look into the "**drive_c**" folder). So I thought it best to keep the "**drive_c**" folder as is and let the "**c:**" link point to it.
I just copied my QPC and qxl.win files to drive F. Now I had qxl.win on my F drive and the qpc executable (qpx2.exe) together with the registry file and smsqe.bin was located in a folder called "**qpc2/Latest**" on drive F. I configured (under windows) qpc so that it went looking for its win1_ in a file called "**qxl.win**" on drive F: (and no longer on drive C: as before). Then I made a link to drive F in the "**dosdevices**" folder (with the drag & drop method described above). Then I changed the name of the link (which was "F") to "f:" so that wine would know that this was to be the F: drive. So far, so good.

Now came the moment of truth. I opened a terminal window and typed the following:
```
wine /windows/F/qpc2/Latest/qpc2.exe
```
This is the name of the program (wine) followed by the path to the windows program to be executed.
This started the qpc2.exe file (i.e. QPC2) on my windows drive F in the qpc2/Latest folder. QPC started up. I noticed that I couldn't get at the config menu by keeping the shift key depressed, but that wasn't really a problem. QPC loaded all of my boot file from the qxl.win drive and booted normally, but slowly (see below). However, when I got to the flashing cursor (after everything had booted alright, and I boot a lot of software), and typed something, the cursor disappeared and the machine hung.
I restarted QPC and got it to crash again a bit later, even though sometimes it wouldn't crash immediately. Let's just say that the behaviour was erratic. Definitely not encouraging. However, to cut a long story short, I did get it running by copying the QPC folder back into the "drive_c" folder and then starting it with the following command line:
```
wine    /home/Wolf/.wine/drive_c/qpc2/Latest/
qpc2.exe
```
Strangely enough, even though this was the same QPC, it worked. No crashes, or at least not often (but there are still some unexplained crashes). I must also say that I only tried it in full screen mode at first. Later attempts at having it run in window mode failed abysmally. But, there again, as I always run QPC in fullscreen mode, I didn't mind.

What I did mind was the apparent speed: it was so s-l-o-w! At some stages, under the PE, you could even see individual items being drawn! I played around some and had just decided to forget having QPC on Linux when, as a last sort of test, I compiled some largish program of mine. To my surprsie, that actually seemed to go pretty quckly.
So I switched back to windows to do some (empirical!) tests; using the following routines:

```
100 DEFine PROCedure test1
110 LOCal a
120   a=DATE
130   FOR lp=1 TO 100000
140     PRINT lp
150   END FOR lp
160   a=DATE-a
170   PRINT "time : ";a;" secs"
180 END DEFine test1
190
280 DEFine PROCedure test2
290 LOCal a%,b%,lp,a
300   a=DATE
310   FOR lp=1 TO 100000
320     a%=RND(1 TO 10)
330     b%=RND(1 TO 20)
340     a%=a%+b%
350   END FOR lp
360   a=DATE-a
370   PRINT "time : ";a;" secs"
380 END DEFine test2
```

Under Windows, test2 would give an average of 4 seconds, and test 1 an average of 143 seconds. I then ran them under "wine". test2 also gave an average of 4 seconds, but Test1 was so slow that I changed the loop in line 130 from 100000 to 1000 - and go something like 67 seconds on average. That would have put the whole test at probably 6700 seconds against QPC's 143. Unacceptable, but at least I had identified some part of the problem, it lies in the routines writing to the screen.

I had also notced that "wine" had protested that it couldn't create a 16 bit drawing surface. I checked my colour resolutions setting using the Linux "Yast" program and, true enough, I was running the Linux desktop in 24 bit mode.

I switched to superuser mode and reduced that to 16 bits and switched back. Running the same test1 as above, gave me 19 seconds, so something like 1900 seconds if I had used the enitre 100000 iterations. Now this is still more that 10 times slower than QPC under Windows. It still feels, well, not fast but at least the terrible slowness I first experienced was gone. So that is something I can probably get used to.

Also, I was using the bog standard Linux screen drivers. I know that there is a special Linux screen drivers around for my video card, but I hadn't downloaded and used them (yet). Doing so made no difference, unfortunetaly.

Finally, I copied the command line used above (`"wine /home/Wolf/.wine/drive_c/qpc2/Latest/qpc2.exe`") into a simple text file I called wine.sh. I then rightclicked on that file and made it executable. So now I only have to click on it from within Konqueror to start QPC.

Anyway, I now got a working QPC under linux which doesn't crash often -and when it does, it generally is right at the start. I'll try to find out when/why the crashes occur, and let you know!

ZIP and UNZIP

by Dilwyn Jones

Zip is a utility program for packing a set of files into a single file, known as an "archive", with the capability of compressing files. The term "compressing" means reducing the space taken by those files, although zip can store files uncompressed in an archive if it thinks that attempting to compress those files may result in a file which is actually larger than the original.

In simple terms, what Zip does is to try to reduce the storage space needed for a set of files and to combine them into a single large file. This is useful for distributing programs via websites and bulletin boards, for example, because downloading a single file is more convenient than downloading several individual files, and if the archive is smaller than the sum of the individual file sizes, then of course you are online for shorter periods and so your telephone bill benefits too.

Zip is the program which does the packing and compressing.

Unzip is the program which decompresses and unpacks the files back into their original form.

INFO-ZIP

This is the name of the organisation which co-ordinates the Zip and Unzip rograms for various computing platforms such as Linux, Windows, RISC OS, Amiga, Atari and of course QDOS. Info-Zip is a diverse, internet based workgroup of about 20 primary authors and over 100 beta-testers, originally formed in 1990 in the USA. The Info-Zip programs are basically free to use and copy, subject to the terms of the licence (basically inclusion of Info-Zip copyright notices within redistributions), and you can get hold of the Zip/Unzip source code files if you wish to see how the software works.

INFO-ZIP for QDOS

One of the "team of twenty" is our own Jonathan Hudson, who ported the version for QDOS and SMSQ/E systems. At the time of writing, the version numbers for the official QDOS versions were Unzip version 5.41 and Zip version 2.3. You may also come across version 5.32 of Unzip which is a perfectly usable version in my experience.

It is important that you use the Info-Zip versions of Zip and Unzip. Older versions by various authors exist, but these have limitations such as inability to handle level 2 directories, being slower than the current version and the possibility of creating zipped files which work only on the system on which that file was created. Info-Zip is a cross-platform system which means that for people like Jonathan who create files on one system and move them to other systems they stand a good chance of working on that other system. So stick to the Jonathan Hudson versions of the QL Zip and Unzip programs.

Other Archivers

There are other archiving programs out there for various systems, including QDOS. You may come across LHA, Zoo, and RAR for example - versions of these by various authors exist for the QL. You can get most of these programs from my website at

www.dilwyn.uk6.net/arch/index.html

Broadly speaking, Zip is the most common archiver for the QL these days, as you will see if you visit any website or bulletin board offering QL software for download. I suggest you start off using Zip and Unzip and explore the others later as your confidence in using such programs grows.

Such archivers are generally "portable" - files compressed on Windows, say, can generally be decompressed on a QL. The only areas where you may encounter some difficulty are with QL executable programs when they are unzipped on a non-QL system causing loss of the job headers, or with filename extension separators. The QL prefers '_' between parts of filenames, other systems generally prefer '.' - although QL Zip and Unzip cope automatically with any conversion requirements where needed.

Command Lines

Zip (and Unzip) is what is generally known as a "command line" utility. In other words, you type in commands to make it do what you want. Modern programs are generally menu driven or pointer driven to make them easier to use, and Zip on the QL is no exception. You can get "front end" programs like Archivers Control Panel to make Zip and Unzip (as well as other archiving programs) easier to use and we'll look at these in the next part of this series.

Obtaining Copies of ZIP and UNZIP

Since Info-Zip is freely copyable within the terms of the licence, you can get hold of copies on disk or CD from PD libraries, or by downloading copies from websites. Usually, the package will include everything - the programs themselves, the documentation files and sometimes the source code files (usually the sources are available separately). The list of files is quite large and you may not be interested in the source code files, for example, so you will need to know which parts you need to keep.

If you are downloading copies from the web, for example, you will generally find that Zip and Unzip themselves come to you in a zipped format! This means you have to have a copy of Unzip already to decompress them, thus putting you in a bit of a sticky situation if you do not already own a copy! Fear not, Jonathan Hudson has kindly made the distribution of Unzip available in a format which will "self-extract" as long as you follow the instructions to the letter.

Here are some web page addresses from where you can get copies of Zip and Unzip. Note that Zip comes in two forms - you can get a version with file encryption facilities and a version which does not. The latter is generally smaller and is good enough for most purposes unless you specifically need encryption facilities.

Jonathan Hudson's website:

http://www.daria.co.uk

On this site, scroll way down the page until you reach the section for Zip and Unzip (see Figure 1 on the next page) and click on the underlined link for the package you wish to download.

The first link is for downloading a copy of Unzip. This will download a file called UNZIP541xQ.BIN which a self-extracting version of QDOS Unzip v5.41, which is basically a zipped copy of the Unzip files, but with a small amount of extra code to help it extract the files without needing an existing copy of Unzip.

The third and fourth links are to download whichever version of the Zip package you prefer to use. With these you will get files called ZCR23xQ.ZIP (for the version with encryption) or ZIP23xQ.ZIP.Once you have a copy of Unzip, you

QUANTA

Independent QL Users Group

World-wide Membership is by subscription only,
offering the following benefits:
Bimonthly Newsletter – up to 40 pages
Massive Software Library – All Free!
Free Helpline and Workshops
Regional Sub-Groups. One near you?
Advice on Software and Hardware problems
Subscription just £14 for UK members
Overseas subscription £17

Barclaycard: Visa: Access: MasterCard: Accepted

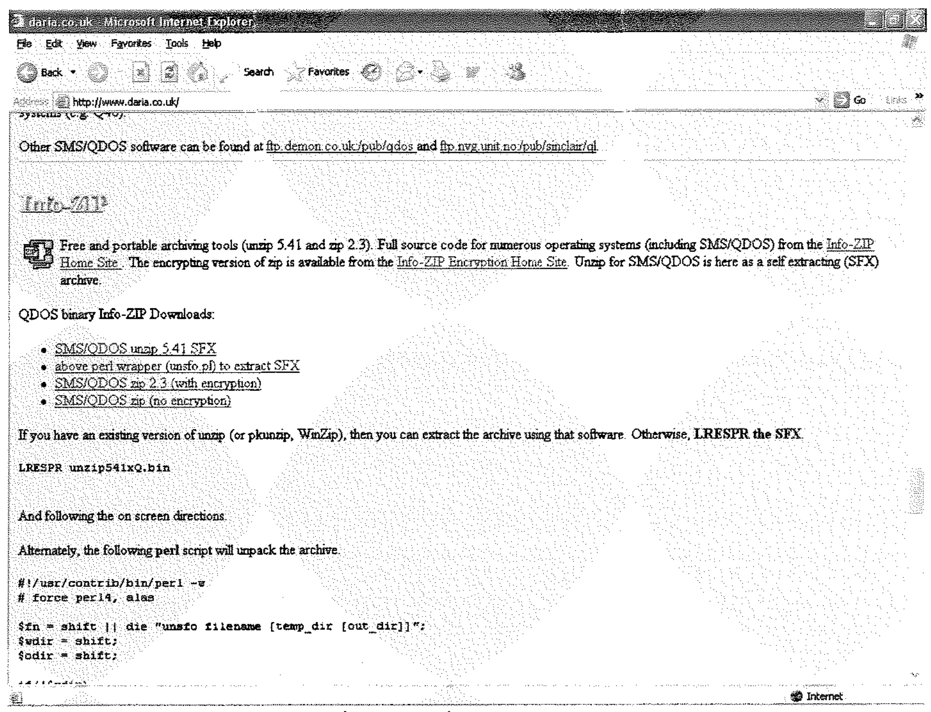
Now in our Twenty Fourth Year

Further details from the Membership Secretary

John Gilpin, 181, Urmston Lane
Stretford, Manchester, M32 9EH (UK).
Tel. +44 (0) 161 865 2872

Or

Visit the New Quanta Web Site
http://www.quanta.org.uk
E-mail: membership@quanta.org.uk



daria.co.uk - Microsoft Internet Explorer

Address http://www.daria.co.uk/

Other SMS/QDOS software can be found at ftp.demon.co.uk:/pub/qdos and ftp.nvg.unit.no:/pub/sinclair/ql.

Info-ZIP

Free and portable archiving tools (unzip 5.41 and zip 2.3). Full source code for numerous operating systems (including SMS/QDOS) from the Info-ZIP Home Site. The encrypting version of zip is available from the Info-ZIP Encryption Home Site. Unzip for SMS/QDOS is here as a self extracting (SFX) archive.

QDOS binary Info-ZIP Downloads:

- SMS/QDOS unzip 5.41 SFX
- above perl wrapper (unsfo.pl) to extract SFX
- SMS/QDOS zip 2.3 (with encryption)
- SMS/QDOS zip (no encryption)

If you have an existing version of unzip (or pkunzip, WinZip), then you can extract the archive using that software. Otherwise, **LRESPR the SFX**.

```
LRESPR unzip541xQ.bin
```

And following the on screen directions.

Alternately, the following **perl** script will unpack the archive.

```
#!/usr/contrib/bin/perl -w
# force perl4, alas

$fn = shift || die "unsfo filename [temp_dir [out_dir]]";
$wdir = shift;
$odir = shift;
```

Figure 1 - Zip/Unzip on Jonathan's website

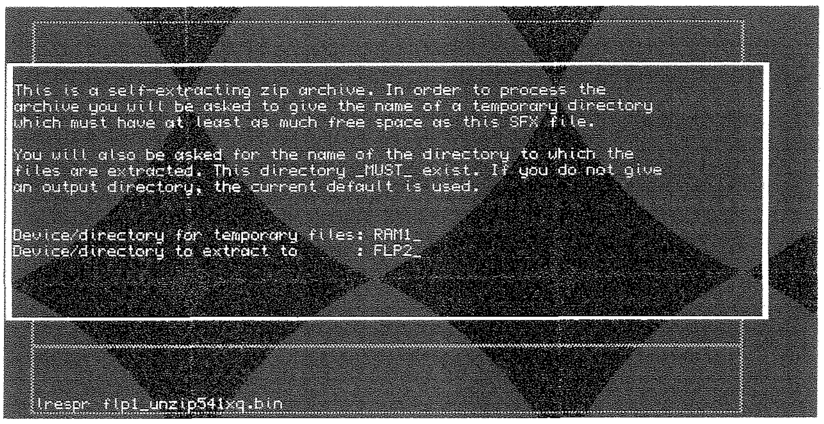
can use it to decompress either of these packages.
If for any reason you cannot get copies from Jonathan's website, you can get copies from the official Info-zip website at www.info-zip.org or you can use their FTP (File Transfer Protocol) site at:
ftp://ftp.info-zip.org/pub/infozip/
If all else fails, try my website's Archivers page at:
www.dilwyn.uk6.net/arch/index.html

Installing ZIP and UNZIP

Assuming you downloaded these packages on a PC, transfer the archives to a QL or to your emulator and I'll explain how to unpack them ready for use.
The first one to unpack is Unzip, for obvious reasons - you'll need it to unpack the Zip package!
As explained above, UNZIP541xQ.BIN is a Self Extracting file for the QL. Ensure no other programs are running (not even the hotkey job if you are using pointer environment - a HOT_STOP command will stop that if needed). You will need Toolkit 2 active on your system - if you are using SMSQ/E the commands needed are already built into your system, and most QL systems with disk drives these days have Toolkit 2 built in. Some interfaces such as a Gold Card may need a command such as TK2_EXT to activate the toolkit commands, if your boot program does not already do this.
Your system needs to have expanded memory too, at least 384K of RAM, but the more the merrier! It means that you cannot unpack on a demo version of QemuLator with just 128K of RAM, for example.
Ideally, your system will have a ramdisk (most systems do these days).
Version 5.41 of Unzip will state that it needs something called 'signal extensions' which is a small toolkit of job communication extensions by Richard Zidlicky. You can usually find this in PD libraries or on the web in a zip file called SigExt30_zip. If you have these extensions, fine, but Unzip will work perfectly well without them, although version 5.41 of Unzip will issue a message saying that the extensions are missing. Ignore the message.
I'll assume that the UNZIP541xQ_BIN file is on a disk in FLP1_ and that you will be decompressing it to FLP2_ and using RAM1_ for temporary files created during the unpacking process. If you only have a single disk drive, you can unzip to a ramdisk temporarily and copy the files to an empty floppy disk later.
To start with, issue an LRESPR command on the file:

```
LRESPR "FLP1_UNZIP541xQ_BIN"
```

If it has not been renamed to a QL standard filename with '_' characters in it, the filename may



```
This is a self-extracting zip archive. In order to process the
archive you will be asked to give the name of a temporary directory
which must have at least as much free space as this SFX file.

You will also be asked for the name of the directory to which the
files are extracted. This directory _MUST_ exist. If you do not give
an output directory, the current default is used.

Device/directory for temporary files: RAM1_
Device/directory to extract to      : FLP2_

lrespr flp1_unzip541xq.bin
```

Figure 2 - The 'SFX' screen

be FLP1_UNZIP541xQ.BIN, it does not really matter but if it ends with '.' and it is being processed on a QDOS system, you should put the filename in quotes as QDOS will not allow '.' characters in unquoted filenames.
The screen shown in figure 2 will now pop up, asking you to enter the temporary files device and the device to extract to. Enter RAM1_ for the first question and FLP2_ for the second (assuming these are the drives being used of course).

The program will now create the temporary files on RAM1_ and issue the following message:

```
[ To extract the files run the command LRUN RAM1_SFX_bas ]
[ Press any key to exit.                                 ]
```

Press a key for this part of the program to finish, then as prompted type this command into BASIC:

```
LRUN RAM1_SFX_bas
```

In most cases, this will automatically complete the unzipping process by itself. On some systems (and my QPC2 v3.32 system seems to be one of these) it fails to run properly and gives an error message, in which case you'll have to intervene manually and type in these commands (which is essentially what the SFX_bas program contained) yourself:

```
EW RAM1_SFX_EXE;'-d FLP2_'
delete ram1_SFX_exe
delete ram1_SFX_dat
delete ram1_SFX_bas
```

Hopefully you will now have the required files on FLP2_ and the temporary files will have been cleared out of the ramdisk by the delete commands.
If you cannot get the "self-extract" procedure to work at all even with the above help, you will need to obtain a copy of unzip on disk from somewhere, or unzip the files in Windows or Linux and copy them to a QL disk. This is not recommended as the executable file header will be lost (Unzip will give the error Bad Parameter when you try to execute it) and some heavy duty programming needed to restore them, like this.

```
100 fl = FLEN(\"flp2_unzip")
110 base = ALCHP(fl)
120 LBYTES FLP2_Unzip,base
130 DELETE flp2_unzip
140 SEXEC Flp2_Unzip,base,fl,51810
```

What this little program does is to load what is left of the program and uses the SEXEC command to save the executable, adding a dataspace value of 51810 to 'repair' the program.

Files Needed

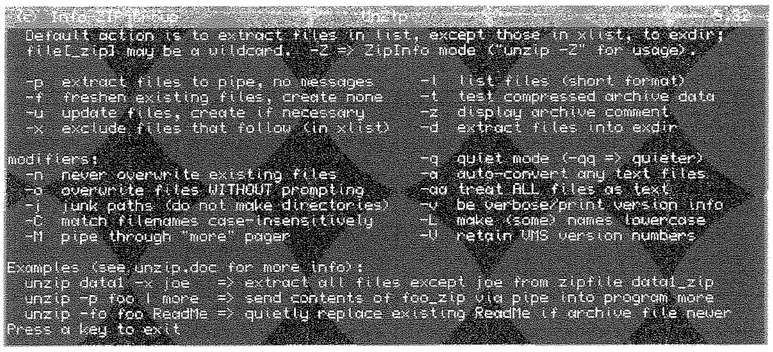
Basically, you need all the programs and documentation files. The main programs are the files called Zip and Unzip. There are various other utilities and all sorts of documentation files - read these when you get a chance although some may appear a bit too technical at this stage for less experienced users.
On my system, I've thrown all the unzip package files into a directory called win1_unzip_, all of the Zip package files into win1_zip_ and put copies of Zip and Unzip into the directory where I keep copies of all my QL programs, win1_exec_

Getting Started

Before we make any real use of either program, I need to point out that the programs have a somewhat limited screen of help information built in - to see this, just execute the program with no parameters:

```
EXEC FLP1_UNZIP
```

This will display the help screen shown in figure 3.



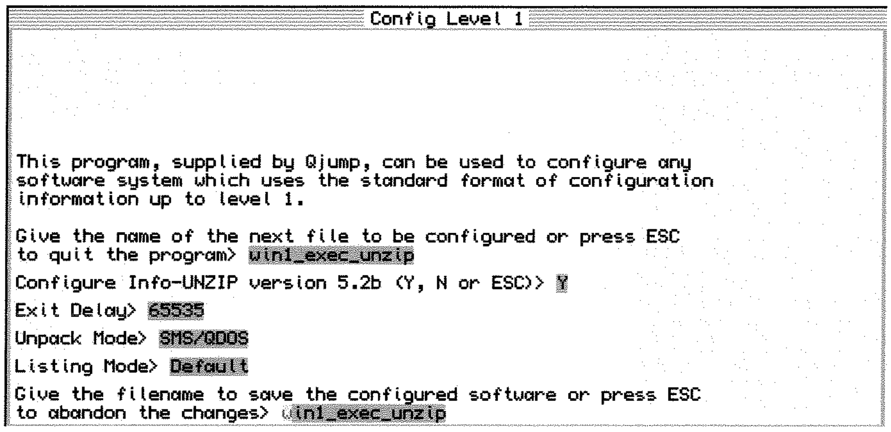
```
 Default action is to extract files in list, except those in xlist, to exdir;
 file[_zip] may be a wildcard.  -Z => ZipInfo mode ("unzip -Z" for usage).

 -p  extract files to pipe, no messages      -l  list files (short format)
 -f  freshen existing files, create none     -t  test compressed archive data
 -u  update files, create if necessary       -z  display archive comment
 -x  exclude files that follow (in xlist)    -d  extract files into exdir

modifiers:                                   -q  quiet mode (-qq => quieter)
 -n  never overwrite existing files          -a  auto-convert any text files
 -o  overwrite files WITHOUT prompting       -aa treat ALL files as text
 -j  junk paths (do not make directories)    -v  be verbose/print version info
 -C  match filenames case-insensitively      -L  make (some) names lowercase
 -M  pipe through "more" pager               -V  retain VMS version numbers

Examples (see unzip.doc for more info):
 unzip data1 -x joe   => extract all files except joe from zipfile data1_zip
 unzip -p foo | more  => send contents of foo_zip via pipe into program more
 unzip -fo foo ReadMe => quietly replace existing ReadMe if archive file newer
Press a key to exit
```

Figure 3 - Unzip's built in help screen

It is also important to know that Unzip has a configuration block built in, which lets you configure some aspects of how the program behaves. 3 options can be set for Unzip:
1. Timeout for the 'press any key' display before Unzip finishes. This is expressed as 65535 (which means wait forever for a keypress), 0 (which means don't wait at all) or a value from 1 to 32,767 in units of 1/50th of a second to wait for a keypress before giving up and finishing anyway. A value of 50 means 1 second, while the highest value of 32,767 allows for a delay of about 660 seconds.

2. Unpack mode: this can be SMS/QDOS (filenames use '_' characters) or default value (probably non-QDOS - "." character in filenames)
3. Listing mode - can be default (non-QDOS) or specifically SMS/QDOS.

Figure 4 shows the configuration process. You can use either the Config or MenuConfig programs to alter these settings for Unzip. You will be able to set Zip as well once we have unpacked that program.



```
                         Config Level 1

This program, supplied by Qjump, can be used to configure any
software system which uses the standard format of configuration
information up to level 1.
Give the name of the next file to be configured or press ESC
to quit the program> win1_exec_unzip
Configure Info-UNZIP version 5.2b (Y, N or ESC)> Y
Exit Delay> 65535
Unpack Mode> SMS/QDOS
Listing Mode> Default
Give the filename to save the configured software or press ESC
to abandon the changes> win1_exec_unzip
```

Figure 4 - the configuration process

Hopefully, we now have a working copy of Unzip, so we can go about unpacking a working version of Zip.
You should have downloaded a file called either ZIP23xQ.ZIP or ZCR23xQ.ZIP, it doesn't really matter which you got, since the only difference is that the second is a version with file encryption or 'cloaking' facilities and is slightly larger.
To decompress the package you need to use the EW command to pass a small list of parameters to Unzip to tell it exactly what to do:

```
EW FLP1_UNZIP;'-d FLP2_ FLP1_ZIP23xQ.ZIP'
```

The -d FLP2_ part tells it to unzip the file called FLP1_ZIP23xQ.BIN to disk drive FLP2_. If, like me, you only have a single disk drive on your computer, you can use -d RAM1_ instead and have it unpacked to ramdisk RAM1_, then later copy all of the files to a floppy disk. If you have a hard disk system, you can extract the files direct to a directory on that:

```
MAKE_DIR WIN1_ZIP_
EW FLP1_UNZIP;'-d WIN1_ZIP_ FLP1_ZIP23xQ.ZIP'
```

You will have noticed how unwieldy these unzip commands can be - this is what I meant by describing the programs as "command line" utilities. You have to use not very memorable commands to achieve anything, which is why programs like Archivers Control Panel will come in useful later, although if you can't remember the commands, it's possible to write small BASIC programs to help you use Unzip and Zip.
Once Zip has been unpacked, you can use Config or MenuConfig programs to configure the defaults for Zip. There are only 2 options with Zip, one sets the timeout value as for Unzip, while the other tells Zip which is the file type number representing a directory. On most modern QL systems (99% of systems!) it will be 255 for level 2 devices, although older systems such as Thors may use type 3 or 4.

Using Unzip

Now for the fun part, actually using Unzip. In all cases, I'll assume we'll be using it to unpack a file called FLP1_EXAMPLE_ZIP which we have downloaded from the web.
Unzip normally (unless you explicitly tell it otherwise) unpacks files to the DATA_USE default drive. On a floppy disk system, it is usually FLP1_ and on a hard disk system WIN1_, although of course you can set it to just about anything with a DATA_USE command, so if you want to unzip to RAM3_ just issue a DATA_USE RAM3_ command before the Unzip command:

```
100 DATA_USE RAM3_
110 EW FLP1_UNZIP;'FLP1_EXAMPLE_ZIP'
```

Line 100 sets where the unzipping will be sent to, and line 110 starts the unzip program, telling it to unzip FLP1_EXAMPLE_ZIP. Note how the filename is specified as a string after a semi-colon, this is how these option commands are passed to programs like Unzip using EX or EW commands. Note: although some versions of the EXEC and EXEC_W commands have been extended to allow use of these command parameters, it is normally better to use the Toolkit 2 EX and EW versions.
It is possibly to explicitly tell the program where to unzip to using a "-d destination" command. Zip has all sorts of these one letter commands available, preceded by a hyphen. -d states which drive/directory to unzip to:

```
EW FLP1_UNZIP;'-d RAM4_ FLP1_EXAMPLE_ZIP'
```

This will tell unzip to unpack EXAMPLE_ZIP from FLP1_ to ram4_

It is also possible to view a list of what files are contained in EXAMPLE_ZIP, using a -l command (l for listing):

```
EW FLP1_UNZIP;'-l FLP1_EXAMPLE_ZIP'
```

This is useful if you think you may not need to unpack every file. You can get a list and write down those files you think you'll need and just extract those, by specifying the names after the name of the archive. For example, suppose we only want to extract prog1_bas and prog2_bas files from EXAMPLE_ZIP:

```
EW FLP1_UNZIP;'-d RAM4_ FLP1_EXAMPLE_ZIP
prog1_bas prog2_bas'
```

Note that these names may need to match the case of those stored inside the archive. If it says they are called prog1_bas and you enter PROG1_BAS it may not work! There may also be problems such as filenames with names clashing with QDOS directory names, for example. Some of these filename clash problems can be hard to diagnose and not always easy to solve, so be aware of this as a possible cause of a mysterious problem you might run into at some stage!
Unzip also supports '*' and '?' wildcard options to selectively extract and process files. * stands for any sequence of characters, and ? stands for any single character. So if you wanted to extract only files ending with _bas (BASIC programs) you could try this:

```
DATA_USE RAM4_
EW FLP1_UNZIP;'FLP1_EXAMPLE_ZIP *_bas'
```

Note: you could have used either _bas or .bas, it seems to recognise both.

And if there were several versions of a program, e.g. prog1_bas, prog2_bas, prog3_bas and so on, you could use prog?_bas as a wildcard to extract all of them.

```
EW FLP1_UNZIP;'FLP1_EXAMPLE_ZIP prog?_bas'
```

Difficult to grasp at first, but you get used to it after a while.
Normally, if a file already exists, you get a 'yes/no' prompt offering to overwrite it. It is possible to bypass this and overwrite without asking using a -o command to overwrite without asking - useful (if dangerous!) if you know you will need to overwrite a large number of files without having to answer yes or no every time.
As well as specifying which files are to be unzipped, you can also tell it to specifically exclude specified files using a -x command. For example, if you don't want the text files (those ending with _txt) you can use a command like this:

```
EW FLP1_UNZIP;'FLP1_EXAMPLE_ZIP -x *_txt'
```

A Program to Unzip

I mentioned you could write a BASIC program to control Unzip. This is a little easier than you might think. Here's an example to extract all files from a specified zip file:

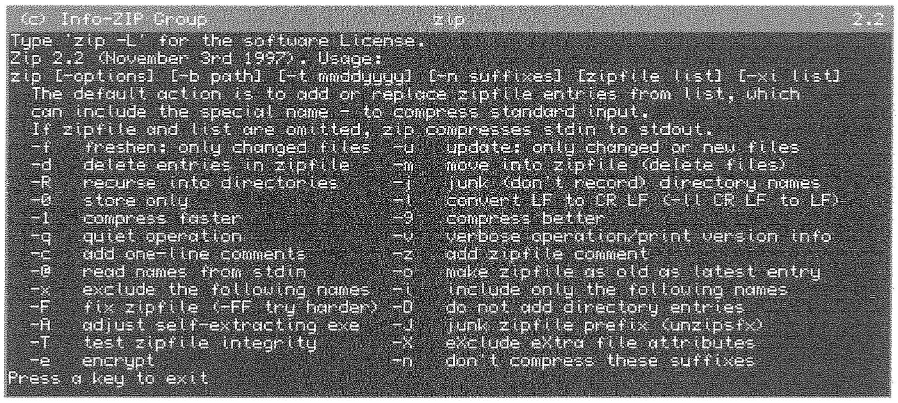
```
100 CLS : CLS #0
110 INPUT #0,'File to unzip › ';ip$
120 INPUT #0,'Unzip to › ';op$
130 INPUT #0,'Where is UNZIP program ›
    ';uz$
140 EW uz$&'unzip';'-d '&op$&' '&ip$
```

Zipping Files

The process of packing files into a single compressed file is called Zipping. Like unzipping, we can put all files from a single place into an archive, or use wildcards or specify a list of files to be included. If an archive already exists, files are added automatically to it, otherwise a new zip file is created from scratch.

```
EW FLP1_ZIP
```

entered by itself, this command displays Zip's built in help screen, see figure 5.



```
(c) Info-ZIP Group                    zip                                2.2
Type 'zip -L' for the software License.
Zip 2.2 (November 3rd 1997). Usage:
zip [-options] [-b path] [-t mmddyyyy] [-n suffixes] [zipfile list] [-xi list]
  The default action is to add or replace zipfile entries from list, which
  can include the special name - to compress standard input.
  If zipfile and list are omitted, zip compresses stdin to stdout.
  -f   freshen: only changed files  -u   update: only changed or new files
  -d   delete entries in zipfile    -m   move into zipfile (delete files)
  -R   recurse into directories     -j   junk (don't record) directory names
  -0   store only                   -l   convert LF to CR LF (-ll CR LF to LF)
  -1   compress faster              -9   compress better
  -q   quiet operation              -v   verbose operation/print version info
  -c   add one-line comments        -z   add zipfile comment
  -@   read names from stdin        -o   make zipfile as old as latest entry
  -x   exclude the following names  -i   include only the following names
  -F   fix zipfile (-FF try harder) -D   do not add directory entries
  -A   adjust self-extracting exe   -J   junk zipfile prefix (unzipsfx)
  -T   test zipfile integrity       -X   eXclude eXtra file attributes
  -e   encrypt                      -n   don't compress these suffixes
Press a key to exit
```

Figure 5 - Zip help screen

To pack files into an archive, use a command like this:

```
EW FLP1_ZIP;'FLP1_EXAMPLE_ZIP FLP2_*'
```

This puts all files it can find from FLP2_ into the zip file called FLP1_EXAMPLE_ZIP. If FLP1_ already existed, it added all the files to it, otherwise it created FLP1_EXAMPLE_ZIP and put

zipped copies of the files on FLP2_ into it. Note - it does not delete the original files on flp2_, it puts compressed copies of them into the zip file (although there is a -m option to MOVE files into an archive instead!).
You can replace the FLP2_* wildcard with a list if you prefer:

```
EW FLP1_ZIP;'FLP1_EXAMPLE_ZIP FLP2_BOOT
FLP2_MYPROG_BAS'
```

As with Unzip, you can use -x commands to specifically exclude some files if required.
If you need to, you can delete specified files from an archive, e.g. when you accidentally inserted a file you did not wish to be included:

```
EX     FLP1_ZIP;'FLP1_EXAMPLE_ZIP     -d
myprog2_bas'
```

That command deleted myprog2_bas from the archive called FLP1_EXAMPLE_ZIP.
You can include a comment into an archive by using a -z command. This comment is just that - a comment among the list of files, useful for including short copyright notices, for example:

```
EX FLP1_ZIP;'FLP1_EXAMPLE_ZIP -z'
```

The -z command causes the zip program to ask you for a comment. It can be more than one line long. Enter each line and press ENTER. Simply enter a full stop on the last line to end the comment. So, using -z you could add a copyright notice including your name and address, ending the comment with a full stop on the last line (that line is not part of the comment):
The zip program prompts with the line:
enter new zip file comment (end with .):
You would then enter the comment:

```
This program is copyright Fred Bloggs 2006
123 The Street
Anytown
AB12 3CD
.
```

If the file already contains a comment, it shows you what it was so that you know what you are replacing. To see and test the comment you have just added, use a simple unzip listing:

```
EX FLP1_UNZIP;'-l FLP1_EXAMPLE_ZIP'
```

The comment appears before the list of files.
If you are in the habit of using sub-directories, a useful option is a -R command, which causes Zip to recurse into sub-directories, which in simple terms means include files contained in sub-directories as well as the current directory:

```
EX FLP1_ZIP;'FLP1_EXAMPLE_ZIP -R RAM4_*'
```

That command tries to add all files in RAM4_ and all sub-directory contents on that drive into FLP1_EXAMPLE_ZIP. Be careful, it is possible to include a huge number of files by mistake. For example,

```
EX FLP1_ZIP;'FLP1_EXAMPLE_ZIP -R WIN1_*'
```

would try to add my entire hard disk content into a zip file on FLP1_, some hope!
Zip normally needs to make some temporary files while it is working. By default it will create these on RAM1_ but you can make it create them on another drive with the -b option:

```
EX FLP1_ZIP;'-b WIN1_ FLP1_EXAMPLE_ZIP
RAM2_*'
```

That command created temporary files on WIN1_ while it tried to add all files from RAM2_ into FLP1_EXAMPLE_ZIP.
The commands -1 through to -9 offer a trade off between compression efficiency and speed. -9 maximises compression at the expense of speed (although on modern fast systems it might not make much difference to speed!).
There are all sorts of other options, but I won't over complicate matters here. At first, you'll just be using Zip and Unzip to compress and decompress all files on given drives to or from given zip files. Once you are confident with the simplest use, you can start to explore the more advanced options.

Creating Self-Extracting ZIP files

There is a program called makesfx which adds some code to the start of a zip file to make it into a self-extracting file. Unfortunately, it is not particularly easy to use or very reliable. If you are still interested, read the QDOS-specific part of the Zip documentation.

In the next part of this article, we'll take a look at Archivers Control Panel and a few other front end systems for Zip and Unzip.

Co-Prozessing

by Stephen Poole

For decades, the size and speed of new models of processors have been following 'Moore's Law'. That is, the number of transistors doubles every 18 months. Processor-speed also increases regularly and now reaches some 4Ghz, and all that thanks to continuous miniaturisation. As circuits get smaller, electrons travel less distance, so if you halve the distance, you double the speed. But from physics we know that all conductors have resistance and that resistance heats them up. The larger the conductor, the better heat is evacuated. So we reach a point where reducing processor size causes chip overheating, requiring cooling fans and radiator-panels. A reasonable compromise can be found at 3.5Ghz, and such a pause in speed development should drastically reduce chip prices...

Market research shows that more laptops are being sold than desktops, and this tendancy is accelerating. To get above 3.5Ghz requires enclosing processors in refrigerated units, hardly feasable in laptops. So getting extra speed requires a different strategy. Sinclair implemented multi-tasking on the QL, and the QL had not only serial communications, and the network but also the 8049 coprocessor.

Multitasking involves rapid switching between concurrent tasks, using scheduling routines to find which tasks can have a slice of processor time according to their priorities, and polling routines which check all peripherals for interrupt requests. So it becomes clear that the QL kernel is basically the hub of a multi-tasking, multi-processor system, in the sense that all peripherals, including memory, are basically independant devices. One of the beauties of the QL is that right from the start it had a simple optimised operating system capable of handling all this in a very logical way.

In 1984, Sinclair adopted one solution to the speed problem. To ramp up the QL, they included the 8049 coprocessor to offload the main 68008 which was relieved of input/output, keyboard scanning and beeping etc. In the same way, some QL systems handled floating-point coprocessors, as indeed early PCs had sound cards, and modern PCs have graphics cards.

Early calculators were analogical, that is to say that their functions were hard-wired, so there were no synchronisation delays, therefor they were very fast. Then appeared hybrids, which had their minimal operations hard-wired, the whole being coordinated by digital-memory processors. Indeed this is how modern computers work. The 68000 processor has a limited number of instructions hard-wired into the silicon chip, all the other instructions being assembled into memorised programs.

Right from the start, computer companies realised the need for communication between processors. If two chips are identical, how can they communicate? One solution is to cut data up into small numbered packets and send them down a cable, and to sort and reassemble them at the other end. But which chip gets control? An early solution was that each chip would generate a random number. The one with the smallest number would get priority. Otherwise, one chip could be declared master and the other slave. This means that the master-chip needs to do some scheduling from time to time, so it spends some time multi-tasking before each chip gets its independant task to chew on.

Again it is clear that coprocessing also contains elements of networking, such as when you have many processors all looking for work or waiting to communicate partial results, when there must be some chip that has master-status so as to be able to arbitrate. So every chip must have a clock and all those clocks must be regularly synchronised: More overheads!

Back in the seventies most manufacturers had their own solutions to these problems and agreed to at least one standard...the Graphics Kernel System. The PETRI system was invented to simplify and help standardise designs, defining circuits in terms of knots, branches, vectors and tokens. This allowed computer-aided program writing and circuit design. So parallel processing is a compromise of analogic instructions sets, digital buffers, and synchronisation. Very early on in the fifties, parallel programs were already written using the FORTRAN language, which has continuously been modernised since, having been designed so that code could be ported to most machines.

The biggest problem has been how to parallelise serial programs. One solution is to emulate coprocessors by using the equivalent number of multi-tasking jobs, having broken down programs into independant modules. But most computer algorythms are inherently serial, having been conceived to run on serial machines! Take, for example, the human brain: It is fed by hundreds of thousands of nerve-inputs, which interact in massively-parallel regions of cortex, feeding in turn thousands of output nerves. Man has spent centuries building serially-operating calculating machines, as copying the complexity of the brain is beyond our current comprehension and technology. Yet even the giant 'Thinking Machine' supercomputer had 65536 parallel processors, giving an instantaneous collective speed measured in Teraflops, (Thousands of Gigaherz of floating-point operations per second). But these machines may spend collectively a large part of their time internally thrashing (with their chips waiting to communicate).

All this may sound unfamiliar, but even now you can vectorise up to 16 PC's, just running under Linux! And even Atari produced a 20Mhz transputer called the 'ATW' way back in the early eighties, which universities could operate in parallel mode. Perhaps the greatest pioneer of all was Seymour Cray who many times held the world processing speed record, and who even planned a DNA chip!

Why write of Coprocessing? Well, both Intel and AMD have already started selling double and quadruple processors, and it is clear that this coprocessing is the way of the future. I for one would love to access the graphics card on my PC under QPC right now, and the next step would be to use QPC2 to write parallel programs for multi-processor PC computers, on which it can already run.

But what are parallel programs like? There are several tendancies, but the most interesting type uses tree-access methods. Coprocessors load to and from interconnecting interacting trees. Supervision is assured by mesh-networking techniques originally using programs like ADA, (based on Pascal, which run Military communication networks). Indeed, future computing may well use the massive cooperative internet mode of computers, as with Citizen Science projects or massively multiplayer games sessions.

The main problem with coprocessing is the lack of an analytical mathematic theory to automate the translation of serial programs to parallel ones. (We are still largely stuck with the Theories of queues and pipelines). Important early work was done by the Frenchman J. Ichbiach, inventor of the innovative language called 'GREEN'. The general adoption of the Unix system was very important in standardising methods. One relatively simple technique is to convert program data into array-format, which is immediately compatible with parallel computer programs.

Current research shows that DNA chips may well be built into research computers soon, and that the study of how genes interact to construct proteins could help us to understand how to reprogram DNA to produce hyper-fast molecular-sized circuits. This research will open new perspectives. Further off could be Quantum computers, where bits may be simultaneously on and off, but the system theory for these has yet to be written.

In the meantime, try rewriting a few of your own programs in a modular way, and then compile each part using Turbo to make them virtual Coprocessor-compatible. This is the way of the future...

But then there is always lazy coprocessing: You load Word into one processor and Excel into the other....and the whole lot still just sits there waiting for input! Vive le Progrès!

Monday Morning QL'ing Guide

by Dilwyn Jones

A quick guide to QL terminology's real meanings. Or alternatively "How My QL Feels On A Monday Morning."

ADATE - short for Any Date (except the one you want)

Arc - sometimes known to happen with very old power supplies.

Array - very simple and powerful method of reducing free memory value returned by FREE_MEM.

Algorithm - writing how a routine should work. Writing it out on paper takes the first 90% of the programming time, getting it to work takes the other 90% of the time.

Alphameric - the result of trying to say 'Alphanumeric' first thing on a Monday morning.

AUTO - allows random line numbers to be generated.

Babbage - invented computers and then discovered the First Law of Computers - they never work properly unless you put QDOS or SMSQ/E on them.

TF Services

Compswitch

A UK 4-way trailing socket designed to switch off computer peripherals automatically when the computer is switched off, or (in the case of an ATX computer) when it auto-powers down. *Compswitch* has one control socket, and three switched sockets.
Can be used with lights/hifi/monitors—ie a QL monitor can be used as a switch control.

Cost £24

superHermes

A major hardware upgrade for the QL

All Hermes features (working ser1/2 at 19200, independent baud rates/de-bounced keyboard/ keyclick) IBM AT kbd I/F // HIGH SPEED RS232 at 57600// serial mouse port and 2 other RS232 inputs// 3 I/O lines // EEPROM

Cost (including manual/software)	**£90** (£92/£93)
IBM AT UK layout Keyboard	**£11** (£13/£15)
Serial mouse	**£8** (£8.50/£9)
Capslock/scrollock LED	**£1** (£1.50/£1.50)
Keyboard or mouse lead	**£3** (£3.50/£3.50)
High speed serial (ser3) lead	**£4** (£4.50/£4.50)

Hermes available for £25 (£26/£27) **Working ser1/2 and independent input, debounced keyboard.**

SuperHermes LITE: All Hermes features (see above) + an IBM AT keyboard interface only.
Cost (incl keyboard lead) .... **£53** (£54/£55)

QL REPAIRS (UK only)

Fixed price for unmodified QLs, excl microdrives. QLs tested with Thorn-EMI rig and ROM software.

£27 incl 6 month guarantee

Minerva

The ORIGINAL system operating system upgrade

OTHER FEATURES COMMON TO ALL VERSIONS

DEBUGGED operating system/ autoboot on reset of power failure/ Multiple Basic/ faster scheduler- graphics (within 10% of lightning) - string handling/ WHEN ERROR/ 2nd screen/ TRACE/ non-English keyboard drivers/ "warm" fast reset. V1.97 with split OUTPUT baud rates (+ Hermes) & built in Multibasic.

First upgrade free. Otherwise send £3 (+£5 for manual if requd). Send disk plus SAE or two IRCs

MKI...**£40** (£41/£43) MKII...**£65** (£66/£67)

MINERVA RTC (MKII) + battery for 256 bytes ram. CRASHPROOF clock & I²C bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

QL RomDisq

Up to 8 mbyte of flash memory for the QL

A small plug in circuit for the QL's ROM port (or Aurora) giving 2, 4 or 8 mbytes of permanent storage - it can be thought of as a portable hard disk on a card, and reads at some 2 mbytes per second.
Think of it - you could fully boot an expanded QL, including all drivers/SMSQ etc off **RomDisq** at hard disk speed with only a memory expansion needed.

2 mbytes RomDisq	**£39** (£40/£41)
4mbytes RomDisq	**£65**(£66/£67)
8 mbytes RomDisq	**£98** (£99/£100)
Aurora adaptor	**£3** (£3.50/£4)

MPLANE

A low profile powered backplane with ROM port

A three expansion backplane with ROM port included for RomDisq etc. Aurora can be fitted in notebook case and powered off single 5V rail - contact QBranch for details. Two boards (eg Aurora and Gold Card/Super Gold Card/Goldfire fixed to base. Suitable for Aurora (ROM accessible from outside) & QL motherboard in tower case. Specify ROM facing IN towards boards, or OUT towards back of case.

Cost .... **£34** (£35/£36)

I2C INTERFACES

Connects to Minerva MKII and any Philips I²C bus

Power Driver Interface 16 I/O lines with 12 of these used to control 8 current carrying outputs (source and sink capable)

2 amp (for 8 relays, small motors)	**£40** (£43/£44)
4 amp total (for motors etc)	**£45** (£48/£50)

Relays (8 3a 12v 2-way mains relays (needs 2a power driver) .... **£25** (£28/£29)

Parallel Interface Gives 16 input/output lines. Can be used wherever logic signals are required........... **£25** (£27/£28)

Analogue Interface Gives eight 8 bit analogue to digital inputs (ADC) and two 8 bit digital to analogue outputs (DAC). Used for temp measurements, sound sampling (to 5 KHz), x/y plotting........................ **£30** (£31/£32)

Temp probe (-40°C to +125°C)	**£10** (£10.50/£11)
Connector for four temp probes	**£10** (£10.50/£11)
Data sheets	**£2** (£2.50/£3)
Control software & manual (for all I/F)	**£2** (£2.50/£3)

QL SPARES

Keyboard membrane	**no longer on sale**
1377 PAL	**£3** (£3.50/£4)
Circuit diagrams	**£3** (£3.50/£4)
68008 cpu or 8049 IPC	**£8** (£8.50/£9)
8301/8302 or JM ROM or serial lead	**£10** (£10.50/£11)
Power supply (sea mail overseas)	**£12** (£19/£23)

Other components (sockets etc) also available

Prices include postage and packing (Airmail where applicable) Prices are: UK (Europe /Rest of world). Payment by cheque drawn on bank with UK address,/postal order or CASH! I can no longer accept card payments as UK only does PDQ transaction. SAE or IRC for full list and details 22 Feb 03

29 Longfield Road, TRING, Herts, HP23 4DG
Tel: +44 (0) 1442-828254 Fax/BBS: +44 (0) 1442-828255
tony@firshman.co.uk http://www.firshman.co.uk

Backup - that copy you never got round to making before the system crash.

BASIC - Programming language where on the whole it takes less time to learn than the programs do to run. Alternatively, a means of giving even the novice programmer a means to write bad programs.

BEEP - at least it's louder than a Spectrum.

Binary - simple method of scrambling up your data.

Bit - the small part of the occasional program which actually works.

BORDER - marks maximum distance to which you can throw your PC out of the house when it annoys you.

Bug - mistake in your program disguised by totally meaningless error messages.

Byte - see bit. Program where more bits work.

C - Faster way of creating unreadable programs which only (sometimes) work. More error messages can be generated more quickly than SuperBASIC.

Call - self destruct mechanism for SuperBasic programs calling machine code routines.

Character Set - list of symbols used to fill in naughty words in the average programmer's vocabulary.

CLEAR - throw out issues of Quanta and QL User from 1984.

Clock - part of the computer designed to mock its lack of speed.

Cobol - a business language designed to enhance your long term career prospects because nobody understands it, therefore you must be a brilliant programmer if you do.

Coercion - computer forces you to do what it wants, not vice versa.

Compatible - Not a word we really understand when it comes to talking about other computers.

Compiler - program to rearrange your lovely program, scramble it all up and often make it run even less well than it did before.

Conditional statements - "Can't" or "Won't"

CONTINUE - stuff the error, carry on regardless.

Cursor - programmer with a deadline.

Data - vital variables which combine to ensure you have no free memory. See also ARRAY above.

Database - more information than can be run on maximum memory system.

Database Management - they take the flak when the database always fails.

Debug - extracting the dead spider from the mdv1_ slot.

Delete - the art of accidentally erasing your only copy of a file.

DIM - generate a random list of random data of random usefulness.

Dump - where the PC ends up after the latest Windoze crash.

Hex Dump - foul language of programmers when their programs fail miserably.

Edit - "there must be something meaningful in memory if only I could find it."

Emulate - run the same program on another system, only more slowly.

Encode - render something forever unreadable.

Error - standard output from most programs.

Error messages - attempt at making average program output vaguely meaningful.

EXIT - describes how programmers occasionally escape the keyboard for a few seconds every day.

Expert System - someone who actually (just about) understands the computer. On second thoughts, someone who understands what the programis meant to do, not necessarily what it actually does.

File - an unstructured, undocumented, overkill portion of data guaranteed to give your computer (and you) a headache at the sight of just the first byte.

FILL - how a computer's memory fills up with random and usually meaningless data despite being a fully expanded system.

Floating point - a dot that's never where it ought to be.

FORMAT - method of accidentally erasing your only copy of that favourite program.

Fortran - high level programming language enabling the cleverest of scientists to totally corrupt a system without even needing to call in outside help.

Function - part of a program designed to return an incorrect value and generally make it impossible to debug.

Gosub - take your QL underwater.

Goto - method of making your program totally unreadable and even harder to debug. Last resort when Structured Programming doesn't work.

Hard copy - the evidence that your program does not even remotely work.

Hardware - the part of a computer totally unable to do anything without software.

Hexadecimal - A powerful way to conceal your activities.

High level language - simple way to write convoluted programs.

IF - "can't" or "won't"

Incompatible - thing of the terms "hardware upgrade" and "device drivers"

Initialise - scramble up the program before it's even started.

Input - computer didn't create enough of a mess by itself, needs extra help from the user.

Interpreter - part of your computer designed to try to understand what the heck your program is meant to do.

Keyword - subset of English language understood by your QL.

LBYTES - method of overwriting vital part of computer's memory.

Library - a collection of subroutines designed to make your program bigger without even influencing how it runs.

Lisp - computer language designed to test the programmer's grip on reality.

Local - a variable designed to add confusion to your programs. Alternatively, a place to retire to when you give up on getting that program to work.

Logic - used by humans to try to (convincingly) explain why programs don't work.

Loop - add a 'y' and it describes how some programmers work.

Machine code - programming language used when you want to look clever.

Memory - that which you never have enough of to run the latest program on.

Network - never quite understood why the spelling isn't Notwork.

Null - so that's why the program failed to print.

ON...GOTO - method of making your program flow jump to completely unforeseen places.

OPEN - fetch data which is even more random than the stuff that's already in memory.

Operating System - the software most likely to crash a computer (except the QL, maybe)

Parameters - list of usually incorrect values passed to a routine.

Parity - state which described how neither you nor the computer know what to do, so you toss a coin instead.

Password - the security word or number which, for it to work, is so hard to remember it gets written on a sticky note affixed to the computer.

PAUSE - "it's crashed...hasn't it?"

PEEK - look into forbidden places.

POKE - ingenious method of crashing a computer.

Pointer - usually appears on screen just when you realise you've mislaid that cordless mouse.

Press Any Key To Continue - Just like an airline pilot, don't bother reading the instructions, just press all the buttons until you find one which looks like it might make the plane land successfully.

Printer_dat - an attempt to make printers and users speak the same language. Occasionally.

Prototype - program which is usually even worse than the one you eventually get.

QPC2 - valiant effort at making PCs useable by QLers.

Quill - method of writing something just as slowly as its namesake.

Quotes - should always occur in pairs in print statements and rarely do.

RECOL - make the colour scheme even worse than it is now.

Recursion - so that's why my program doesn't work. Usually describes the programmer's language when he/she can't get it to work.

Redundancy - 90% of most programs.

RENUM - try to make orderly sense of the rubbish the user typed in.

REPEAT - execute the same faulty code many times.

Reserved words - some 4 letter words which have to be edited out of transcripts of what the programmers actually said.

RETRY - have another go at generating yet another error.

RETurn - Give up, go and try another part of the program.

RS232 - method of making computers (slowly) refuse to talk to each other.

SAVE - I can't get it to work, so I'll try again tomorrow.

SCALE - make even more of the picture fall off-screen.

SCROLL - instructions for that old program.

SDATE - set the date to an even more random number than now.

Seek time - the time taken to find the right floppy disk.

SEXEC - Not even going there ;-)

SIN - unwaranted use of SEX EC

SMSQ/E - attempt at making QDOS programs run with fewer errors. At least it has more error messages built in.

Sort - method of scrambling the data in a list into the wrong order.

Stack - long list of bugs in a typical program.

STOP - time to go to bed after a long programming session.

Storage - number of boxes needed to store all the bits of a typical expanded QL system.

String - data which should mean something but never really does.

String array - powerful technique for building up long lists of faulty data to fill up way too much memory.

STRIP - not going there either ;-)

Structured - program where the GOTOs and GOSUBs actually do vaguely what they were intended to do.

Subroutine - part of a program you cobbled together at the last minute.

Super Gold Card - description of credit card needed to buy the latest QL hardware.

Syntax Error - the computer gave up trying to understand what the heck the programmer intended it to do.

System Disc - the only (formatted) disc which seems to have any space for your data.

Unsupported - Set of features which your program needs and which nobody has.

Utility - a program so useless it doesn't fit into any other category.

Variable - data which is usually any value except the one your program expects.

WIDTH - measure of how long you spend at your computer, sum of sedentary time, fizzy drinks and junk food consumed.

WINDOW - emergency exit for PC when it's upset Dilwyn.

Write Protected - posh way of saying you forgot to put the disc in the drive.

Zero - dividing by this usually screws up computers.

Compiling QPTR with TURBO

by George Gwilt

Introduction

From version 4.21 Turbo will compile programs containing machine code extensions to S*BASIC which either alter their parameters or require arrays as parameters. Of course Turbo is well known for producing fast compiled code although needing the programs it compiles to be written according to stricter rules than for S*BASIC. Will this, or should this, necessity be an important stumbling block to the wider use of the enhanced Turbo?

To help answer this question I will explain the six steps that I took to arrive at a compilation of QPTR's demonstration program "demo_bas". Users of QPTR will, of course, be aware that its manual stresses that many of its keywords cannot be compiled by Turbo. However, this restriction was removed in Turbo v4.21 so "demo_bas" should provide a useful test of the updated Turbo.

The amended working version of demo_bas is given later.

Steps

Step 1 - Try It

The first obvious step was:

```
LOAD qptr_Demo_bas
EX parser_task
```

What happened?

A diagnostic file was produced indicating that there were well over 100 errors! Some of these were due to my forgetting to install the QPTR extensions, but by no means all. It was obvious that the main problem was that DATA lines contained items that were not just pure numbers or strings.

Step 2 - Alter DATA Lines and Try Again

I should explain that the values in DATA statements are stored in a special way near the start of a program compiled by Turbo. The type of each item is indicated by its first word as described below.

a. If the first word of an item is between 0 and 32766 this is taken as the length of a following string.
b. Values from -1 to -4095 are taken to be negated exponents of floating point numbers, so a four byte mantissa follows.
c. Values from -4098 to -32766 represent integers. The actual integer is
   - value - 7096
d. The value -32767 signals the end of data.
e. The value +32767 signals that the following word is an integer. So this allows all the other integers not represented in c above.

All values in DATA statements are extracted in Parser_task's first pass through the target program to be compiled. So, because of this and also because of the way the DATA values are stored, it is not possible for these values to be defined in terms of variables in the program. Turbo actually goes further and forbids any calculations in DATA items, even though they may theoretically be carried out. Thus even 30*4 is disallowed.

PUT and GET

Examination of the code for demo_bas showed that the DATA lines were set in blocks which were READ by adjacent S*BASIC procedures or functions. Thus the first block was the set of standard attributes to be set in an array later to be used as a parameter in two S*BASIC functions. No block of DATA lines was used more than once.
I decided that it would be sensible to replace the DATA and READ commands by PUT and GET. My first thought was to PUT the data items into a file from which subsequent GETs could extract the information needed. If a note were kept of the starting position in the file of each new block as it was written, the first GET for each block could set the appropriate position when needed, rather like using RESTORE to set the required line of DATA. I then realised that I might as well write each new block of information over the previous block and dispense with keeping a note of the starting position in the file of each block since that would always be at the beginning of the file. Finally I decided to use a pipe instead of a file. Each block would be PUT into the pipe which would be emptied by the corresponding GETs. In this case of course there is no need for positioning.
Applying Parser_task to the amended program still brought up errors, though considerably fewer than at the previous attempts. This time most of the faults were due to using the same name for different types of variable resulting in the message "Ambiguous Names".

STEP 3 - Eliminate Ambiguous Names

Turbo uses a Vector Table to store information about all names used in a compiled program. Each name is allotted its own four-byte space in the table. For integers and floating point number variables the entry is simply a pointer to the value. For strings and arrays, the entry is a pointer to a descriptor which itself has a pointer to the actual values as well as detailing the number and size of dimensions. Turbo allows redimensioning provided that the number of dimensions remain the same. This means that the descriptor for an array variable is always the same size, since the size depends only on the number of dimensions, not their value. The descriptors can therefore be held in a fixed space. This is one of the constraints which helps make for fast compiled programs.
A consequence of this is that if a variable is defined anywhere as an array, it cannot be defined anywhere else except as an array with the same number of dimensions. Turbo will signal these errors with:

"Ambiguous declaration of name."

The cure is easy. Different names must be used for the offending items.
The variables involved in demo_bas are the arrays "ldef%" and "lptr". Both are defined in some places as one dimensional and in other places as two dimensional. I replaced the one dimensional ldef%s by aldef% and the one dimensional lptrs by alptr.

Step 4 - Investigate BAD PARAMETER

The cause of the message "Bad parameter" was that the call to MK_LIL in the function RD_LOTN contained the string lsk$. All strings are dimensioned by Turbo, and this was no exception. Hence the string parameter to MK_LIL was a dimensioned string.
It is very peculiar, but nevertheless it is true, that MK_LIL complains if that string parameter is not a simple, undimensioned, string. In the original demo_bas lsk$ is undimensioned. I should explain that in S*BASIC the entry in the Name Table for a dimensioned string will have a pointer to its array descriptor whereas the entry for an undimensioned string will have a pointer directly to the string itself.
The solution in this case was to ensure that Turbo sent the required string as an expression, which would force it to be undimensioned, and not as a variable, which in Turbo will always be dimensioned. So I replaced lsk$ by lsk$&" wherever it occurred as a parameter. In this case Turbo sets the entry as a pointer directly to the string itself (ie its value) rather than to a descriptor (as for a dimensioned string).
I must say that this particular BAD PARAMETER error signalled by Turbo is due more to the peculiarity of MK_LIL than to any restriction imposed by Turbo itself. After all, if someone tried to use MK_LIL with a dimensioned variable for this string parameter it wouldn't work even if the program was RUN in S*BASIC or compiled by Qlib.

Step 5 - Solve Problems with INTEGERS and FLOATING POINT

Having altered DATA lines and READs to PUT and GET I found that the new program ran correctly in S*BASIC. The compiled version worked well except that the notes produced in the music window were wrong. The reason for this was that the number 2^(1/12), which was defined as "semitone", was set by Turbo as 1. The reason for this is that Turbo, on encountering "2^(1/12)" decides that all of 2, 1 and 12 are integers, which of course they are, but then goes on to deduce that the result of the calculation must also be an integer. Thus 1/12 becomes 0. After that x^(1/12) will be taken as 1 as long as x is an integer.
The solution I used was to define "semitone", which is a floating point variable, as 2 and then raise "semitone" to the power 1/12.

This case is an example of the difference between S*BASIC and Turbo which relates to the way integers and floating point numbers are dealt with. Another example of this relates to PUT.
The command PUT#3,40 results, oddly enough, in the floating point number 40 being sent to channel 3 if the order is given in S*BASIC.
However, if that instruction appears in a program compiled by Turbo it is the integer 40 which is sent to channel 3. It is very important to control just what format will be used by PUT since the contents of the file will be extracted later by GET and set to a variable. That variable must have the same type as the item sent by PUT.
PUT can be controlled by using variables rather than literal numbers. Thus to send the integer 4 one might set

```
  i4% = 4
```

and issue the order

```
  PUT#3, i4%
```

Replacing i4% by i4 would result in a floating point 4 appearing in channel 3.
The first example of this is in lines 12080 to 12100. I give first the original lines and then the ones I replaced them by.

Original Lines

```
12080    DIM std_wattr%(3)
12090    RD_WATT std_wattr%
12100      DATA 0,1,grey,white
```

Replacement Lines

```
12080    DIM std_wattr%(3)
12090      PUT#4,i0%,i1%,i255%,i7%
12100    RD_WATT std_wattr%
```

Here, all the items of DATA are redefined as integers. This is because the numbers have to be entered into an integer array which will later be used as a parameter to one or more machine code extensions.
To show how the READs have been modified I give the original and altered version of the S*BASIC procedure RD_WATT.

Original

```
22290 DEFine PROCedure RD_WATT (lattr%)
22300   READ lattr%(0), lattr%(1),
        lattr%(2), lattr%(3)
22310 END DEFine RD_WATT
```

Altered

```
22290 DEFine PROCedure RD_WATT (lattr%)
22300   GET#3, lattr%(0), lattr%(1),
        lattr%(2), lattr%(3)
22310 END DEFine RD_WATT
```

As you can see, "READ" has been replaced by "GET#3," and this is the case throughout, except for RD_SPRITE where the corresponding DATA lines contain no expressions and can thus be left unaltered.

Step 6 - Array Parameters

Turbo arranges that parameters to Basic routines will be sent by value unless a REFERENCE directive is given just before the DEFINE for that routine. Since all array parameters to Basic routines must be sent by reference, it is necessary to include REFERENCE directives for these. There are six places where this was needed.

Line	Routine
20190	RD_WDEFA
20360	RD_LOTA
21020	RD_AWTA
21700	RD_AOTA
22230	RD_IATT
22290	RD_WATT

Miscellaneous

The variable "compiled" is defined in the original program. Since the altered version is intended to be compiled by Turbo, the Turbo TK extensions must be loaded. One of these is the function COMPILED. Hence the line defining "compiled" has been REMarked out. In any case Parser_task would flag it as an error indicating that you can't assign values to a machine code function. Anyway COMPILED gives a better result than "compiled" since the latter will show that a program run in a daughter basic is compiled when it obviously isn't.
When I now compiled and executed the program it worked until I clicked on the items in the bottom half of the window. I then found that the music window at top right gave odd notes in some places. This was due to there not being enough stack. The solution here is to configure codegen_task to give more than the minimum 350 bytes. I don't know what the smallest safe amount is but 600 seems ample. The command TURBO_objstk can set the stack from 350 bytes up to 2048 bytes so 600 is well within Turbo's capability.

Summary

The question at the beginning of this was whether the rather strict rules of Turbo were a stumbling block to writing fast compilable programs using QPTR.
The program demo_bas was not written with Turbo in mind. Even so there was only one main area where several changes were needed, namely DATA statements.
As I have shown here, the use of PUT and GET to a pipe can replace the DATA statements quite satisfactorily.
In fact, program styles can vary tremendously. It seems to me that a program written from scratch to use QPTR can without difficulty be in a form suitable for compilation by Turbo as well as being capable of being run in S*BASIC. It is just a question of adopting an appropriate style.

Program

The Changed Version of DEMO_BAS

```
1 REMark Compile with no windows open!!
2 :
3 REMark version 1.03 - 13 AUG 05
4 :

*************************************************************************
*                    100 to 200  set Turbo options                      *
*************************************************************************

100 TURBO_locstr "Report":REMark Ignore/Create
110 TURBO_windo  0
120 TURBO_optim  "Brief":REMark Rem/Fast
130 TURBO_model  ">64":REMark >64
140 TURBO_struct "Struc":REMark Freef
150 TURBO_diags  "i":REMark Omit/Include
```

Feeling out on a Limb?
Reach out for QBranch
Suppliers of Quality QDOS/SMSQ products
Hardware and Software

20 Locks Hill
Portslade
Sussex
BN41 2LB
UK
Tel: +44(0)1273 386030
Fax: +44(0)1273 430501
Mobile: +44(0)7836 745501
email : sales@qbranch.demon.co.uk
www.qbranch.demon.co.uk

We can accept payment by Visa, Mastercard, Maestro and Switch. We also accept sterling cheques drawn on a UK bank. or payment direct to our bank:
Natwest
Sort code 60-17-01
Acc. 84534826
BIC : NWBKGB 2L
IBAN :
GB34 NWBK 6017 0184 5348 26

VISA MasterCard Maestro

QDT

The Ultimate QL Desktop
v1.02
£41.50 + p&p
£2.00 UK / 2.50 EU / £ 3.00 RoW
Needs SMSQ/E & extended colours to run

EasyPTR v4

Pts 1&2
Major Rewrite!
£42.00
Produce High Colour pointer-driven menus from SuperBASIC
Also on CD Rom

Text 87 £ 79.00

High Colour Patch (all systems) £9.00
ESC/P2 drivers £ 26.00 / Fountext £39.00
Typset94 / 2488 drivers £29.00 ea.

SMSQ/E and QPC 2

SMSQ/E for Gold Card / Atari / QXL / Q40
Aurora High Colour Driver included with Gold Card Version
Now Only **£15.00 + postage**

QPC 2 v 3.33

Now only £ 42.00

Upgrade Prices:
from v 3.xx Free
from v 2.xx £13.90
from v 1.xx £34.00

Delivered on CD ROM with Manual as Adobe Acrobat File

QPCPrint

Print to any Windows Installed Printer from Your QL Programs !
£27.00

Programs upgraded to high colour version

QMake / WinEd / QPAC 1 / QPAC 2 / QSUP / SuQcess / Disk Mate 5
Send old master disk and £1.60 per disk to cover costs

Free !

Paid Upgrades	
QD 98 to QD 2003	£10.50
QSpread 2001 to QSpread 2003	£10.50

Return Master Disk *with order*

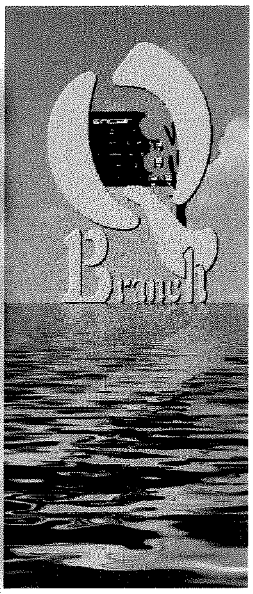
Big Price Drop on SMSQ/E. Any version now only

£15.00!

Coming soon. All versions of SMSQ/E on CD. We are compiling all versions on one CD with full documentation. Since there is extra room on the CD we also intend to include the sources and documentation.

New version of QPC2 out soon.

See you all at the Hove Workshop and Quanta AGM in April.



Programs

Utilities

Fifi2	File Finder	£21.00
QSup	Utilies	£30.00
QSpread	Spreadsheet	£51.00
Cueshell 2	File Manager	£15.00
QPAC 2	File Manager & Utilities Package	£42.00
QPAC 1	Calendar, Clock, Calculator, Sysmon	£22.00
QLoad/Qref	Fast load for QDOS	£15.00
QTYP2	Spell Checker	£31.00
QLQ	Printer Utility	£30.00
SuQcess	Database	£28.00
Q-Route	Route Finder	£25.00
Knight Safe3	Backup Program	£35.00
Fractal Collection	Fractals	£35.00
QCount	Accounting	£25.00

Programming

QD 2003	Text Editor & More	£ 49.00
QBASIC	QLiberater to QD Link	£ 15.00
QLiberator	Basic Compiler	£ 50.00
QD + QBASIC		£ 63.00
QD + QBASIC + QLiberator		£104.00
QPTR	Pointer Toolkit	£ 32.00
MasterSpy	Fast Text Editor	£ 30.00
QMake	Assembler Tools	£ 18.00
QMon/Jmon	Monitor - Upgrade only	£ 22.00
BASIC Linker	Basic Library Linker	£ 22.00
Disa 3	Dissassembler	£ 34.00
QMenu	Menu Extensions & tutorial	£ 16.00
Easyptr v4	Toolkits & Programming Extns	£ 41.50
Easyptr v4	Part 3 C extensions	£ 14.00

HARDWARE

We have a rotating stock of both new and second user hardware. It is best to call or email us for details of what is available.

Recycled Items (when available)

Super Gold Card	£110.00
Gold Card	£ 45.00
Aurora	£ 65.00
QXL	£ 35.00
superHermes	£ 65.00
DiRen Keyboard Interface	£ 15.00
Qplane	£ 5.00

New Items

Aurora	£70.00
Aurora w/8301 & Minerva	£80.00
8mb Rom Disq	£98.00
4mb Rom Disq	£65.00
2mb RomDisq	£39.00
mPlane	£34.00
MCplate	£ 6.50
Various braQuets	£ 8.00
Gold /Super Gold Card Batteries	£10,00

We also have a collection of standard QLs, QL Power supplies and some QL books.

Cables for the Aurora, Qubide and Super Gold Card ROMs and other QL accessories are also available from us.

Call for details

```
160 TURBO_taskn  "pe_demo"
170 TURBO_repfil "ram1_err"
180 TURBO_objdat  40:TURBO_objstk 600
190 TURBO_objfil  "ram1_demo3_task"
200 TURBO_buffersz 200:TURBO_ref

*************************************************************************
*  400 has been REMarked out. (COMPILED is a Turbo TK code extension)   *
*************************************************************************

400 REMark COMPILED=JOB$(-1)<>''
410 REMark IF RMODE=8 THEN MODE 4
420  IF COMPILED: OPEN #1,'con': INK 0: REMark INK ensures that #1 has been used!
430 :
1000 init
1010 :
1020 IF COMPILED
1030   DR_PPOS main_defn,-1,-1,m_lflg%,m_mflg%,m_cty%
1040 ELSE
1050   DR_PULD main_defn,-1,-1,m_lflg%,m_mflg%,m_cty%
1060 END IF
1070 DR_AWDF #mvch,main_defn,0
1080 FOR i=midCn-10 TO midCn-2 STEP 2,midCn+2 TO midCn+10 STEP 2: BLOCK #mvch;nwxs%,1,0,i;green
1090 REPeat 1
1100   RD_PTR main_defn,item%,swnum%,action%,xrel%,yrel%,m_lflg%,m_mflg%,m_cty%
1110   it=item%
1120   IF swnum%<0
1130     SELect ON it
1140       =0:DR_UNST main_defn:CLOSE #mvch:STOP
1150       =1:BEEP 1000,10:beepn= beepn+1
1160          new_beep$= beep1$&beepn&beep2$&FILL$('s',beepn<>1)&'.'
1170          CH_ITEM main_defn,-3,0,-1,'',new_beep$
1180          DR_IDRW main_defn,1
1190       =2:FOR i=0 TO m_nrow-1
1200            FOR j=0 TO m_ncol-1
1210              IF NOT(m_mflg%(i,j)&&16) THEN m_mflg%(i,j)=m_lflg%(2)+1
1220            END FOR j
1230          END FOR i
1240       =3:CH_WIN main_defn:m_lflg%(3)=1
1250     END SELect
1260   END IF
1270   IF swnum%=0 THEN IF (item% DIV 256)=1 THEN BEEP xrel%*100+1,note%(nnote-yrel%)
1280 END REPeat 1
1290 :
10000 DEFine PROCedure init
10010 :
10020 : REMark  Define a general-purpose moveable channel
10030 : REMark  and some useful colours.
10040 :
10050   OPEN_NEW#4,pipe_512:CONNECT 4 TO 3
10060   mvch=FOPEN('con'):INK #mvch;0
10070   black=0:red=2:green=4:white=7:grey=255
10080 :
10090 : REMark  Possible item type codes: retx is added to return
10100 : REMark  when hit, with (retr) or without (retn) a re-draw
10110 :
10112   DIM new_beep$(70),beep1$(30),beep2$(6),hit$(2),do$(2),cancel$(2),help$(2),move$(2),size$(2),x$(40)
10116 :
10120   text=0:sprite=2:blob=4:pattern=6:text%=0
10130   retr=256:retn=-256
10140   hit$=CHR$(1):do$=CHR$(2):cancel$=CHR$(3)
10150   help$=CHR$(4):move$=CHR$(5):size$=CHR$(6)
10160 :
10170   lxs%=6*5:lys%=12:nwxs%=80:nwys%=30:nwys=30
10180   DIM lxs1%(3):FOR x=0 TO 3:lxs1%(x)=4+(lxs%+4)*x
10190 :
10200 : REMark  Set up an array with the best approximations to
10210 : REMark  the correct values for BEEP to produce a scale.
10220 :

*************************************************************************
*     In 10230 semitone is set to 2 to force 2^(1/12) to be fp.        *
*************************************************************************
```

```
10230   nnote=nwys:midCf=261:midCn=nnote/2:semitone=2:semitone=semitone^(1/12)
10240   botnote=midCf
10250   DIM note%(nnote)
10260   FOR i=midCn TO nnote
10270     note%(i)=1/7.1E-5/botnote-10:botnote= botnote*semitone
10280     nn=(i-midCn) MOD 7
10290     SELect ON nn=0,1,3,4,5:botnote= botnote*semitone
10300   END FOR i
10310   botnote=midCf
10320   FOR i=midCn TO 0 STEP -1
10330     note%(i)=1/7.1E-5/botnote-10:botnote=botnote/semitone
10340     nn=(midCn-i) MOD 7
10350     SELect ON nn=1,2,3,5,6:botnote= botnote/semitone
10360   END FOR i
10370 :
10380   beep1$='You have used the BEEP item ':beep2$= ' time'
10390   beepn=0
10400 :

*************************************************************************
*    10410 to 10450 set numbers as integers, 10460 as floating point    *
*                        for PUT and GET                                *
*************************************************************************

10410  im1%=-1:im2%=-2:i0%=0:i1%=1:i2%=2:i3%=3:i4%=4:i6%=6
10420  i7%=7:i9%=9:i10%=10:i12%=12:i14%=14:i16%=16:i18%=18
10430  i20%=20:i22%=22:i30%=30:i40%=40:i60%=60:i64%=64
10440  i70%=70:i75%=75:i80%=80:i84%=84:i132%=132:i150%=150
10450  i160%=160:i230%=230:i232%=232:i248%=248:im=-1:i255%=255
10460  i0=0:i1=1:i2=2:i4=4:i7=7:im1=-1
12000 :
12010 : REMark  The initialisation of the menu itself starts
12020 : REMark  here: this corresponds to a call to WM.SETUP
12030 : REMark  in machine code.
12040 :
12050 : REMark  The standard window attributes
12060 :
12070   RESTORE 12240
12080   DIM std_wattr%(3)
12090     PUT#4,i0%,i1%,i255%,i7%
12100   RD_WATT std_wattr%
12110 :
12120 : REMark  The standard item attributes
12130 :
12140   DIM std_iattr(3,3)
12150     PUT#4,i1,i0
12160     PUT#4,i7,i4,i0,i0
12170     PUT#4,i7,i0,i0,i0
12180     PUT#4,i4,i0,i0,i0
12190   RD_IATT std_iattr
12200 :
12210 : REMark  The data for the pointers
12220 :
12230   main_sprite=RD_SPRT
12240     DATA 6,5,4
12250     DATA '     www     '
12260     DATA '     waw     '
12270     DATA '     waw     '
12280     DATA '     waw     '
12290     DATA 'wwwwwa awwwww'
12300     DATA 'waaaa   aaaaw'
12310     DATA 'wwwwwa awwwww'
12320     DATA '     waw     '
12330     DATA '     waw     '
12340     DATA '     waw     '
12350     DATA '     www     ',''
12360 :
12370   mus_sprite=RD_SPRT
12380     DATA 2,8,4
12390     DATA '    a  '
12400     DATA '    aa '
12410     DATA '    a a'
12420     DATA '    a a'
```

```
12430     DATA '    a   '
12440     DATA '    a   '
12450     DATA '    a   '
12460     DATA ' aaaa   '
12470     DATA 'aaaaa   '
12480     DATA ' aaa    ',''
12490 :
12500   move_sprite=RD_SPRT
12510     DATA 5,4,4
12520     DATA 'aaaaaaaaaa    '
12530     DATA 'awwwwwwwwa    '
12540     DATA 'awwrrrrwwa    '
12550     DATA 'awwrrrrwwaaaaa'
12560     DATA 'awwrrwwwwwwwwa'
12570     DATA 'awwwwwwwwrrwwa'
12580     DATA 'aaaaawwrrrrwwa'
12590     DATA '    awwrrrrwwa'
12600     DATA '    awwwwwwwwa'
12610     DATA '    aaaaaaaaaa',''
12620 :
12630   qjump_blob=RD_SPRT
12640     DATA 0,0,4
12650     DATA '    aaa                         '
12660     DATA '  aaaaaaa                       '
12670     DATA ' aaaa aaaa                      '
12680     DATA ' aa     aa                      '
12690     DATA 'aaa     aaa                     '
12700     DATA 'aa       aa                     '
12710     DATA 'aa       aa      a  a a   a     '
12720     DATA 'aa       aa      a  a aa aa     '
12730     DATA 'aa       aa    a a  a a a a aaa '
12740     DATA 'aa       aa    a a  a a   a a  a'
12750     DATA 'aa       aa    a a  a a   a a  a'
12760     DATA 'aa       aa    a  aa  a   a aaa '
12770     DATA 'aaa   aa aa a  a            a   '
12780     DATA ' aa   aaaa   aa   aaaaaaaa  a   '
12790     DATA ' aaaa  aaaaa    aaaaaaaaaaaa    '
12800     DATA '  aaaaaaaaaaaaaaaaa      aaaaaa '
12810     DATA '    aaa    aaaaaa         aaaa ',''
12820 :
12830   black_patt=RD_SPRT
12840     DATA 0,0,4
12850     DATA 'aaaaaaaaaaaaaaaa',''
14010 :
14020 : REMark  Some loose objects
14030 :
14040   m_nlit=4:DIM m_lflg%(m_nlit-1)
14050     PUT#4,m_nlit
14060     PUT#4,lxs%,lys%,lxs1%(0),i40%,i0%,i0%,cancel$,text+retr,'Quit'
14070     PUT#4,lxs%,lys%,lxs1%(1),i40%,i0%,i0%,'B',text+retr,'Beep'
14080     PUT#4,lxs%,lys%,lxs1%(2),i40%,i0%,i0%,'A',text+retn,'All'
14090     PUT#4,lxs%,lys%,lxs1%(3),i40%,i0%,i0%,move$,sprite+retn,move_sprite
14100   main_lot=RD_LOTA(std_iattr)
14110 :
14120 : REMark  Some information objects
14130 :
14140  i3=3:i1=1:i0=0
14150     PUT#4,i3
14160     PUT#4,i64%,i20%,i0%,i0%,text,red,i3,i1,'Demo'
14170     PUT#4,i132%,i10%,i0%,i20%,text,black,i0,i0,'of the Pointer Toolkit'
14180     PUT#4,i40%,i20%,i80%,i2%,blob,black_patt,qjump_blob
14190   main_iot_1=RD_IOT
14200 :
14210     PUT#4,i1
14220     PUT#4,i230%,i10%,i0%,i0%,text,black,i0,i0,beep1$&beepn&beep2$&'s.'
14230   main_iot_2=RD_IOT
14240 :
14250 : REMark  Some information sub-windows
14260 : REMark  to put the objects in
14270 :
14280     PUT#4,i2
14290     PUT#4,i132%,i30%,i4%,i2%
14300     PUT#4,i0%,i1%,i0%,i7%
```

```
14310     PUT#4,main_iot_1
14320 :
14330     PUT#4,i230%,i10%,i4%,i60%
14340     PUT#4,i0%,i0%,i0%,i7%
14350     PUT#4,main_iot_2
14360   main_iwt=RD_IWT
14370 :
14380 : REMark  One menu sub-window object table
14390 :
14400   m_nrow=4:m_ncol=3:m_nrc=m_nrow*m_ncol:DIM m_mflg%(m_nrow-1,m_ncol-1)
14410     PUT#4,m_nrc
14420     PUT#4,i2% ,i1% ,'U',text%,'Unicorn'
14430     PUT#4,i0% ,i1% ,'T',text%,'Triffid'
14440     PUT#4,im2%,i1% ,'K',text%,'Kryptonite'
14450     PUT#4,i2% ,i4% ,'D',text%,'Dragon'
14460     PUT#4,i0% ,i4% ,'Y',text%,'Yggdrasil'
14470     PUT#4,im2%,i4% ,'O',text%,'Unobtainium'
14480     PUT#4,i2% ,i0% ,'N',text%,'Nematode'
14490     PUT#4,i0% ,i0% ,'A',text%,'Aspidistra'
14500     PUT#4,im2%,i0% ,'B',text%,'Bauxite'
14510     PUT#4,i2% ,im1%,'W',text%,'Wombat'
14520     PUT#4,i0% ,im1%,'J',text%,'Jacaranda'
14530     PUT#4,im2%,im1%,'C',text%,'Chrysolite'
14540   m_aot=RD_AOTA(std_iattr)
14550 :
14560 : REMark  The spacing lists for the
14570 : REMark  menu sub-window
14580 :
14590     PUT#4,m_ncol
14600     PUT#4,i0%,i0%
14610     PUT#4,i60%,i64%
14620     PUT#4,i80%,i84%
14630     PUT#4,i80%,i84%
14640   main_xspc=RD_AST
14650 :
14660     PUT#4,m_nrow
14670     PUT#4,i0%,i0%
14680     PUT#4,i12%,i14%
14690     PUT#4,i20%,i22%
14700     PUT#4,i16%,i18%
14710     PUT#4,i12%,i14%
14720   main_yspc=RD_AST
14730 :
14740 : REMark  The row list
14750 :
14760     PUT#4,m_nrow
14770     PUT#4,m_aot
14780     PUT#4,i0%,i3%
14790     PUT#4,i3%,i6%
14800     PUT#4,i6%,i9%
14810     PUT#4,i9%,i12%
14820   main_rwt=RD_RWT
14830 :
14840 : REMark  The control definition
14850 :
14860   DIM m_cty%(2,2):m_cty%(0,0)= 2
14870   m_cty%(1,0)=0:     m_cty%(1,1)=0:m_cty%(1,2)=1
14880   m_cty%(2,0)=2*6+20:m_cty%(2,1)=0:m_cty%(2,2)=1
14890     PUT#4,i2%,i4%,i7%,i4%
14900   main_ctdy=RD_CDEF
14910 :
14920 : REMark  Some application sub-windows: the
14930 : REMark  second is a menu sub-window
14940 :
14950     PUT#4,i2
14960     PUT#4,nwxs%,nwys%,i160%,i2%
14970     PUT#4,mus_sprite,'C'
14980     PUT#4,im1
14990 :
15000     PUT#4,i232%,i70%,i4%,i75%
15010     PUT#4,i0,'W'
15020     PUT#4,i0,main_ctdy
15030     PUT#4,i0%,i6%
```

```
15040     PUT#4,main_xspc,main_yspc
15050     PUT#4,i0,i0
15060     PUT#4,main_rwt
15070   main_awt=RD_AWTA(std_wattr%)
15080 :
15090 : REMark  The window definition itself
15100 :
15110     PUT#4,i2%,i1%,i255%,i7%
15120     PUT#4,i248%,i150%,i30%,i30%
15130     PUT#4,main_sprite,main_lot,main_iwt,main_awt
15140   main_defn=RD_WDEF
15150 :
15160 END DEFine
15170 :
20000 DEFine FuNction RD_SPRT
20010   LOCal tmp_patt$(32,32)
20020   LOCal xs,ys, xo,yo, md, l, x
20030   xs=0:ys=-1:READ xo,yo,md
20040   REPeat l
20050     READ x$:IF x$='' THEN EXIT l
20060     ys=ys+1:tmp_patt$(ys)=x$&FILL$(' ',32)
20070     IF LEN(x$)>xs THEN xs=LEN(x$)
20080   END REPeat l
20090   l=ALCHP(SPRSP(xs,ys+1))
20100   SPSET l,xo,yo,md,tmp_patt$(0 TO ys,1 TO xs)
20110   RETurn l
20120 END DEFine RD_SPRT
20130 :
20140 DEFine FuNction RD_WDEF
20150   LOCal lattr%(3)
20160   RD_WATT lattr%
20170   RETurn RD_WDEFA (lattr%)
20180 END DEFine RD_WDEF

*************************************************************************
*        20185 declares lattr% to be a one dimensional array            *
*           In 20200 aldef% replaces the original ldef%                 *
*************************************************************************

20185 REFERENCE lattr%(0)
20190 DEFine FuNction RD_WDEFA (lattr%)
20200   LOCal aldef%(3), lspr, lloose, linfo, lappl,a%,x
20210   GET#3, aldef%(0), aldef%(1), aldef%(2), aldef%(3)
20220   GET#3,lspr,lloose,linf,lappl
20230   RETurn MK_WDEF (aldef%, lattr%, lspr, lloose, linf, lappl)
20240 END DEFine RD_WDEFA
20250 :

*************************************************************************
*                  In 20580 lsk$&"" replaces lsk$                       *
*************************************************************************

20260 DEFine FuNction RD_LOT
20270   LOCal nitem, count(3), lattr(3,3)
20280   GET#3,nitem
20290   IF nitem
20300     nitem=nitem-1
20310     RD_IATT lattr
20320     RETurn RD_LOTN
20330   END IF
20340   RETurn 0
20350 END DEFine RD_LOT
20355 REFERENCE lattr(0,0)
20360 DEFine FuNction RD_LOTA (lattr)
20370   LOCal nitem, count(3)
20380   GET#3,nitem: IF nitem: nitem= nitem-1: RETurn RD_LOTN
20390   RETurn 0
20400 END DEFine RD_LOTA
20410 DEFine FuNction RD_LOTN
20420   LOCal item, ltyp, a$(85), lsk$(nitem)
20430   LOCal ldef%(nitem,6), lptr(3,nitem), lstr$(nitem,85)
20440   lsk$=''
20450   FOR item = 0 TO nitem
```

```
20460     GET#3, ldef%(item,0), ldef%(item,1), ldef%(item,2), ldef%(item,3)
20470     GET#3, ldef%(item,4), ldef%(item,5)
20480     GET#3, a$: lsk$=lsk$ & a$
20490     GET#3, ltyp
20500     ldef%(item,6)=ltyp: ltyp=(ltyp MOD 256)/2
20510     IF ltyp
20520       GET#3, lptr(ltyp,count(ltyp))
20530     ELSE
20540       GET#3, lstr$(count(0))
20550     END IF
20560     count(ltyp)=count(ltyp)+1
20570   END FOR item
20580   RETurn MK_LIL (lattr, ldef%(TO, 0 TO 1), ldef%(TO, 2 TO 3), ldef%(TO, 4 TO 5), lsk$&"",
        ldef%(TO, 6), lstr$, lptr(1), lptr(2), lptr(3))
20590 END DEFine RD_LOTN
20600 :
20610 DEFine FuNction RD_IWT
20620   LOCal nitem
20630   GET#3,nitem: IF nitem: nitem=nitem-1: RETurn RD_IWTN
20640   RETurn 0
20650 END DEFine RD_IWT

**************************************************************************
*                   In 20680 alptr replaces lptr                         *
**************************************************************************

20660 DEFine FuNction RD_IWTN
20670   LOCal item
20680   LOCal ldef%(nitem,3), latt%(nitem,3), alptr(nitem)
20690   FOR item = 0 TO nitem
20700     GET#3, ldef%(item,0), ldef%(item,1), ldef%(item,2), ldef%(item,3)
20710     GET#3, latt%(item,0), latt%(item,1), latt%(item,2), latt%(item,3)
20720     GET#3, alptr(item)
20730   END FOR item
20740   RETurn MK_IWL (ldef%, latt%, alptr)
20750 END DEFine RD_IWTN
20760 :
20770 DEFine FuNction RD_IOT
20780   LOCal nitem, count(3)
20790   GET#3,nitem: IF nitem:nitem=nitem-1: RETurn RD_IOTN
20800   RETurn 0
20810 END DEFine RD_IOT
20820 DEFine FuNction RD_IOTN
20830   LOCal item, ltyp, work1, work2
20840   LOCal ldef%(nitem,4), lptr(3,nitem), lstr$(nitem,85)
20850   FOR item = 0 TO nitem
20860     GET#3, ldef%(item,0), ldef%(item,1), ldef%(item,2), ldef%(item,3)
20870     GET#3, ltyp
20880     ldef%(item,4)=ltyp: ltyp=ltyp/2
20890     IF ltyp
20900       GET#3, lptr(0,item),lptr(ltyp,count(ltyp))
20910     ELSE
20920       GET#3, work1
20930       GET#3, work2: work1=work1*256+work2
20940       GET#3, work2: lptr(0,item)=work1*256+work2
20950       GET#3, lstr$(count(0))
20960     END IF
20970     count(ltyp) = count(ltyp) + 1
20980   END FOR item
20990   RETurn MK_IOL (ldef%(TO, 0 TO 1), ldef%(TO, 2 TO 3), lptr(0), ldef%(TO, 4), lstr$, lptr(1),
        lptr(2), lptr(3))
21000 END DEFine RD_IOTN
21010 :
21015 REFERENCE lattr%(0)
21020 DEFine FuNction RD_AWTA (lattr%)
21030   LOCal nitem
21040   GET#3,nitem
21050   IF nitem: nitem=nitem-1: RETurn RD_AWTAN
21060   RETurn 0
21070 END DEFine RD_AWTA
21080 DEFine FuNction RD_AWTAN
21090   LOCal item
21100   LOCal aldef%(3), alptr(2), lsk$(2)
```

```
21110   LOCal lawa(nitem)
21120   FOR item = 0 TO nitem
21130     GET#3, aldef%(0), aldef%(1), aldef%(2), aldef%(3)
21140     GET#3, alptr(0), lsk$, alptr(1)
21150     IF alptr(1)<0
21160       lawa(item)=MK_APPW (aldef%, lattr%, alptr(0), lsk$&"")
21170     ELSE
21180       lawa(item)=RD_MSWA
21190     END IF
21200   END FOR item
21210   RETurn MK_AWL (lawa)
21220 END DEFine RD_AWTAN
21230 :

*********************************************************************
*     21240 to 21460, which contained RD_AWT and RD_AWTN, have been     *
*                    removed, for two reasons.                          *
* 1. RD_AWT and RD_AWTN are not used.                                   *
* 2. RD_AWTN contains a mistake anyway.                                 *
*********************************************************************

21470 DEFine FuNction RD_MSWL
21480   LOCal loff%(1),loptr(4)
21490   GET#3, alptr(2), loff%(0), loff%(1)
21500   GET#3, loptr(0),loptr(1),loptr(2),loptr(3),loptr(4)
21510   RETurn MK_APPW (aldef%(TO 3), aldef%(4 TO 7), alptr(0), lsk$&"", alptr(1), alptr(2),
        loff%(0), loff%(1), loptr(0), loptr(1), loptr(2), loptr(3), loptr(4))
21520 END DEFine
21530 :
21540 DEFine FuNction RD_MSWA
21550   LOCal loff%(1),loptr(4)
21560   GET#3, alptr(2), loff%(0), loff%(1)
21570   GET#3, loptr(0),loptr(1),loptr(2),loptr(3),loptr(4)
21580   RETurn MK_APPW (aldef%, lattr%, alptr(0), lsk$&"", alptr(1), alptr(2), loff%(0), loff%(1),
        loptr(0), loptr(1), loptr(2), loptr(3), loptr(4))
21590 END DEFine
21600 DEFine FuNction RD_AOT
21610   LOCal nitem, count(3), lattr(3,3)
21620   GET#3,nitem
21630   IF nitem
21640     nitem=nitem-1
21650     RD_IATT lattr
21660     RETurn RD_AOTN
21670   END IF
21680   RETurn 0
21690 END DEFine RD_AOT
21695 REFERENCE lattr(0,0)
21700 DEFine FuNction RD_AOTA (lattr)
21710   LOCal nitem, count(3)
21720   GET#3,nitem: IF nitem: nitem=nitem-1: RETurn RD_AOTN
21730   RETurn 0
21740 END DEFine RD_AOTA
21750 DEFine FuNction RD_AOTN
21760   LOCal item, ltyp, a$(85), lsk$(nitem)
21770   LOCal ldef%(nitem,2), lptr(3,nitem), lstr$(nitem,85)
21780   lsk$=''
21790   FOR item = 0 TO nitem
21800     GET#3, ldef%(item,0), ldef%(item,1), a$, ldef%(item,2)
21810     lsk$=lsk$ & a$
21820     ltyp=(ldef%(item,2) MOD 256)/2
21830     IF ltyp
21840       GET#3, lptr(ltyp,count(ltyp))
21850     ELSE
21860       GET#3, lstr$(count(0))
21870     END IF
21880     count(ltyp)=count(ltyp)+1
21890   END FOR item
21900   RETurn MK_AOL (lattr, ldef%(TO, 0 TO 1), lsk$&"", ldef%(TO, 2), lstr$, lptr(1), lptr(2),
        lptr(3))
21910 END DEFine RD_AOTN
21920 :
21930 DEFine FuNction RD_AST
21940   LOCal nitem
```

```
21950   GET#3,nitem: IF nitem: nitem=nitem-1:RETurn RD_ASTN
21960   RETurn 0
21970 END DEFine RD_AST
21980 DEFine FuNction RD_ASTN
21990   LOCal ldef%(nitem+1,1),item
22000   GET#3, ldef%(0,0), ldef%(0,1)
22010   FOR item=1 TO nitem+1:GET#3, ldef%(item, 0), ldef%(item, 1)
22020   RETurn MK_ASL(ldef%(1 TO nitem+1), ldef%(0,0), ldef%(0,1))
22030 END DEFine RD_ASTN
22040 :
22050 DEFine FuNction RD_RWT
22060   LOCal nitem
22070   GET#3,nitem:IF nitem:nitem=nitem-1:RETurn RD_RWTN
22080   RETurn 0
22090 END DEFine RD_RWT
22100 DEFine FuNction RD_RWTN
22110   LOCal ldef%(nitem, 1), ptr, item
22120   GET#3, ptr
22130   FOR item=0 TO nitem: GET#3, ldef%(item, 0), ldef%(item, 1)
22140   RETurn MK_RWL(ptr,ldef%)
22150 END DEFine RD_RWTN
22160 :
22170 DEFine FuNction RD_CDEF
22180   LOCal aldef%(3)
22190   GET#3, aldef%(0), aldef%(1), aldef%(2), aldef%(3)
22200   RETurn MK_CDEF(aldef%(0), aldef%(1), aldef%(2), aldef%(3))
22210 END DEFine RD_CDEF
22220 :
22225 REFERENCE lattr(0,0)
22230 DEFine PROCedure RD_IATT (lattr)
22240   LOCal i
22250   GET#3, lattr(0,0), lattr(0,1)
22260   FOR i=1 TO 3: GET#3, lattr(i,0), lattr(i,1), lattr(i,2), lattr(i,3)
22270 END DEFine RD_IATT
22280 :
22285 REFERENCE lattr%(0)
22290 DEFine PROCedure RD_WATT (lattr%)
22300   GET#3, lattr%(0), lattr%(1), lattr%(2), lattr%(3)
22310 END DEFine RD_WATT
```

JUST WORDS!

NOW ALL FREE!

Just Words! has now made all its programs freeware. That means that you can download and copy them freely, although the copyright remains with the author. What you are not allowed to do is resell them, or modify them in any way.

If you are unable to download from our website, then you will still be able to buy the programs at QL shows for a nominal fee to cover costs.

Also on our website is a comprehensive range of QTYP dictionaries in several languages, a handy help and advice section and the latest QL shows information.

Geoff Wicks, 5b Wordsworth Avenue, Sinfin, Derby, DE24 9HQ, U.K.

Tel: +44 (0)1332 - 271366 *email: gwicks@beeb.net*

Web: http://members.lycos.co.uk/geoffwicks/justwords.htm

GPS and QPC – Part 3

by Hugh Rooms

QPC

Having wired up to a COM port (I had previously installed a board in my PC to give me two extra COM ports) I used Hyperterminal in Windows to look for the data and find out the number of the port in use. It was quite a relief and thrill to see the data pouring out of the circuit, and I found that the port was number 6. It had been far easier than I expected. Up to now.

But how could I get the display? I must admit that up to this point I hadn't thought about QPC, in fact I first tried in Linux and C, but then I remembered that SuperBasic is very good at both mathematics and graphics, and decided to give it a go.

The next hurdle was accessing the serial port in SB. I found the documentation (13)(14) confusing, with its talk of "device" and "SER port" and numbers that could refer to Windows or QPC. Quite likely this has been explained somewhere in QL-Today or Quanta already, but that would have taken a lot of locating, so I used my tried and trusted "poke it with a stick until it squeaks" technique on SER_GETPORT$, SER_SETPORT, BAUD and FOPEN parameters, until it all boiled down to this, (with arbitrary line numbers):-

```
140 SerPort%= 6: REMark I put this, the
                 Windows COM port number,
                 near the start of ..
150              REMark .. the program to
                 change to SerPort%= 1
                 for the laptop version.
...
810 SerPort$= SerPort%: REMark Necessary
    for the FOPEN parameter string
820 BAUD SerPort%,4800: REMark "820 BAUD
    SerPort$,4800" does not always work
830 cs%= FOPEN("srx"&SerPort$&"IA")
```

For FOPEN, the parameters are a string made up of: "srx", a "receive only" serial port name; "SerPort$", the Windows COM port number as a string -- "SerPort%" will not work here as it doesn't get converted to part of the string; I is "ignore flow control" -- well, it works, whatever it means; and A is for the ‹CR›‹LF› at the end of each line.

I had this in a short SB program to work it out, and it was once again a thrill when the data came tumbling on to the screen. From then on it was just a matter of developing the program to the full version here.

Figure 11, The Program

```
110 REMark Set ShowTrack% to 0 for orbits, 1 for
    track
120 ShowTrack%= 0
130 REMark For orbits: set Blobs% to 1 for blobs, 0
    for lines
140 Blobs%= 1
150 :
160 CLOSE
170 REMark Serial Port: 6 for PC, 1 for Laptop
180 SerPort%= 6
190 maxid= 30: REMark highest permitted satellite
    id no.
200 :
210 IF ShowTrack% >0 THEN
220   Minlon=0: Minlat=0: Maxlon=0: Maxlat=0
230   REMark PRINT Minlon,Minlat,Maxlon,Maxlon
240   REMark For displaying track must set min and
      max
250   REMark  lat and lon or call a procedure to
      do so ...
260   ChiCityMap
270   REMark PfdMap
280   :
290   REMark See Jan Jones page 39, 40
300   IF (1+ Minlon+ Minlat+ Maxlon+ Maxlat)= 1
      THEN
310     CLS
320     PRINT \\\"    *** Map limits not set ***":
        STOP
330   END IF
340 END IF
350 :
360 COLOUR_PAL
370 REMark For orbit display:
380 REMark Colours used for spot showing first
    observed
390 REMark position and use of satellite
400 seentint%= 194: usdtint%= 96
410 :
420 REMark ***********************************
430 sim= 1: REMark Set to 0 for receiver input
440 REMark               1 for file input
450 REMark Set dtof to 1 to save raw data to
    win1_sats_data
460 dtof= 0
470 REMark No point in saving already saved data!
480 IF sim <> 0 THEN dtof = 0
490 REMark ***********************************
500 :
510 CLS#0: CLS
520 CSIZE 0,0
530 :
540 REMark Set up the Serial Port if needed
550 IF sim THEN
560   REMark Serial Port simulated by data file
570   cs%= FOP_IN("win1_gps_sats_dat")
580 :
590 ELSE
600   REMark For real time data from receiver
610   REMark SerPort% set to 6 or 1 at start of
      program
```

```
620   BAUD SerPort%,4800
630   SerPort$= SerPort%
640   cs%= FOPEN("srx"&SerPort$&"IA")
650 END IF
660 :
670 REMark Window to display raw data
680 dd%= FOPEN ("con")
690 WINDOW #dd%,550,75,0,525
700 BORDER #dd%,1,9,1
710 PAPER #dd%,36: INK #dd%,0: CLS #dd%
720 :
730 REMark Window to display error and other
    messages
740 REMark in particular, corrupted data
750 de%= FOPEN ("con")
760 WINDOW #de%,400,75,400,45
770 BORDER #de%,1,9,1
780 PAPER #de%,39: INK #de%,0: CLS #de%: REMark
    paper was 36
790 :
800 REMark Delay in seconds between readings
810 REMark to avoid enormous sats_data file
820 INPUT "Delay in seconds",delay
830 :
840 REMark Window for main display of orbits or
    track
850 asprat= .8: REMark aspect ratio: width/height
860 size = 3.5: REMark for early fiddling with
    windows
870 dc%= FOPEN ("con")
880 Mctr= COS(RAD((Minlat+Maxlat)/2))
890 High= 100*size
900 Wide= 137*size*asprat
910 WINDOW #dc%,Wide,High,0,525-High
920 BORDER #dc%,1,9,1
930 INK #dc%,0
940 PAPER #dc%,37: CLS #dc%
950 :
960 REMark ========================================
970 REMark Set up for track display
980 IF ShowTrack% <>0 THEN
990   difflon= Maxlon- Minlon
1000   difflat= Maxlat- Minlat
1010   SCALE #dc%,difflat,Minlon*asprat*Mctr,
       Minlat
1020   :
1030   REMark Lat and Lon grid
1040   LatLonGrid
1050   :
1060   REMark window to show instantaneous track
       and speed
1070   dt%= FOPEN ("scr")
1080   Topdc%= 525-High+3
1090   WINDOW #dt%,137/4*size*asprat,100/4*size,
       285,Topdc%:
1100   BORDER #dt%,1,9,1
1110   SCALE #dt%,200,-100*asprat,-100
1120   PAPER #dt%,37: CLS #dt%:
1130  ELSE
1140   REMark For orbit display
1150   SCALE #dc%,2.2,-1.1*asprat,-1.1
1160   REMark Set up alternative for orbits
1170   SatSky
1180 END IF
1190 REMark ========================================
1200 :
1210 REMark Open a file to collect the data for
     simulation
1220 IF dtof <> 0 THEN
1230   fc%= FOP_NEW(win1_sats_data)
1240 END IF
1250 :
1260 REMark ::::::::::::::::::::::::::::::::::::::::
1270 REMark This was originally the only display,
     put DEF PROC
1280 REMark round it to allow alternative track
     display
1290 REMark Window #dc% already defined as same
     size for both
1300 DEFine PROCedure SatSky
1310   REMark Sky disk
1320   FILL#dc%,1
1330   INK#dc%,29
1340   ELLIPSE#dc%,0,0,1,1*asprat,0
1350   FILL#dc%,0
1360   :
1370   REMark Draw polar plot grid lines
1380   INK#dc%,12
1390   radials : REMark draw the bearings
1400   REMark now draw the elevations
1410   FOR i = 1 TO 3
1420    ELLIPSE#dc%,0,0,i/3,1*asprat,0
1430   END FOR
1440   LINE#dc%,-1.02*asprat,0 TO 1.02*asprat,0
1450   LINE#dc%,0,-1.02 TO 0,1.02
1460   :
1470   REMark Mark points of compass
1480   INK#dc%,9
1490   CURSOR#dc%,1.03*asprat,0,0,-4
1500   PRINT#dc%,"E"
1510   CURSOR#dc%,-1.03*asprat,0,-6,-4
1520   PRINT#dc%,"W"
1530   CURSOR#dc%,0,1.04,-3,-8
1540   PRINT#dc%,"N"
1550   CURSOR#dc%,0,-1.03,-3,2
1560   PRINT#dc%,"S"
1570   :
1580   REMark Mark azimuth scale
1590   FOR i= 30 TO 330 STEP 30
1600     IF (i MOD 90) = 0 THEN NEXT i
1610     j= i+90
1620     CURSOR#dc%,1.03*COS(j*PI/180)*asprat,1.03
         *SIN(j*PI/180),-6,-5
1630     PRINT#dc%,360-i
1640   END FOR i
1650   :
1660   REMark mark elevation scale
1670   FOR i= 0 TO 3
1680     CURSOR#dc%,0,i/3,-6,-5
1690     PRINT#dc%,90-30*i
1700   END FOR i
1710 END DEFine SatSky
1720 :
1730 REMark ::::::::::::::::::::::::::::::::::::::::
1740 :
1750 REMark Display set up, now get ready for GPS
     data
1760 :
1770 REMark Array to store raw data lines from
     receiver
1780 DIM rawdata$(5,128)
1790 REMark Array to store satellite data ..
1800 REMark with:  IdNo, Bearing, Elevation, Usage
1810 REMark where: Usage is 0 for not used, 1 for
     used.
1820 DIM satsvis(11,3) : REMark Allow for 12
     satellites in view
1830 :
1840 DIM satsusd(11)   : REMark List of Ids of
     satellites used.
1850 :
1860 REMark I use copies of the raw data ...
```

```
1870 DIM satlist$(6,128):REMark Satellite data from
     $GPSGV input
1880 REMark Allow for 7 $GPSV lines (OTT -- never
     more than 2!)
1890 :
1900 DIM rmcdata$(128) : REMark Lat, Long, Time
1910 :
1920 REMark Posns for orbits plotted as lines
1930 REMark to draw a blob for first point, then a
     line
1940 REMark Store: Old x,y; New x,y: 5th item, 0=
     new orbit ..
1950 REMark .. so draw a blob, 1= orbit started so
     draw line
1960 DIM posns(maxid,5): REMark need enough for
     each satellite
1970 REMark set all to zero for a start
1980 FOR i%= 0 TO maxid-1
1990   FOR j%= 0 TO 4
2000     posns(i%,j%)= 0
2010   END FOR j%
2020 END FOR i%
2030 :
2040 CLS
2050 REMark For plotting track, need a POINT to
     start
2060 REMark then continue with lines.
2070 REMark Don't bother not defining this if
     orbits
2080 FirstPt%=0
2090 :
2100 lines=0:      REMark A count of input lines
2110 GoodLines=0: REMark Another count of input
     lines
2120 CLS #dd%: REMark I needed this for raw data
     display
2130 :
2140 REMark ======================================
2150 REMark Setting up complete, now for ...
2160 REMark ... data reading and display
2170 :
2180 REMark Repeat loop for continous display
2190 REMark delay at end
2200 REMark Each run through loop deals with a
     single set
2210 REMark of data sentences, five in my case,
     starting
2220 REMark with a $GPGGA. Sent each second, but
     read
2230 REMark at a rate determined by missing some
     with the delay
2240 REMark Test for end in gpsdata$ function
2250 :
2260 REPeat orbits
2270   AT #dd%,0,0: AT 0,0: REMark To keep the
       display tidy
2280   REMark I read all the sentences at one go so
       that I
2290   REMark don't end up with some from a later
       second's lot.
2300   rawdata$(0)= gpsdata$("$GPGGA"): REMark Wait
       for a first
2310   rawdata$(1)= gpsdata$("$GPGSA"): REMark ...
       read the rest
2320   rawdata$(2)= gpsdata$("$GPGSV")
2330   rawdata$(3)= gpsdata$("$GPGSV")
2340   rawdata$(4)= gpsdata$("$GPRMC")
2350   lines= lines+5: REMark lines, GoodLines
       bit messy ...
2360   REMark ... intended it to help look at
       corrupted data
2370   REMark Check lines for not corrupted data
2380   FOR i= 0 TO 4
2390     REMark Go through data to look for
         corrupted lines.
2400     REMark This became a long winded process,
         so that is
2410     REMark why I read all the sentences in one
         go.
2420     REMark Ignoring all the set of data, if
         there is one
2430     REMark corrupted item ,is a bit OTT, but
         safe.
2440     :
2450     REMark Sometimes get lines much longer
         than spec. ..
2460     REMark .. 'Long' lines seem to be a normal
         line but ..
2470     REMark .. no <CR><LF> and followed by
         more, ..
2480     REMark  .. badly formed, data, so ignore
         them:
2490     ChkFld%= '*' INSTR rawdata$(i)
2500     IF LEN(rawdata$(i)) > ChkFld%+ 3 THEN NEXT
         orbits
2510     :
2520     IF dtof <> 0 THEN
2530       PRINT #fc%,rawdata$(i): REMark Save in
           file if needed
2540     END IF
2550     :
2560     REMark Ckeck 'Checksum' field
2570     Check%= CheckSum(rawdata$(i))
2580     IF Check% <> 0 THEN
2590      NEXT orbits
2600     END IF
2610     GoodLines= GoodLines+1
2620     DisLine dd%,GoodLines&" "&rawdata$(i),0
2630   END FOR i
2640 :
2650 REMark Get ready forlist of satellites used
2660 REMark from $GPGSA data line
2670 gpsAdata$ = rawdata$(1): REMark cautiously use
     copies
2680 :
2690 REMark Get ready for list of satellite data
2700 REMark First $GPGSV line gives no of $GPSV
     lines
2710 satlist$(1) = rawdata$(2)
2720 REMark Extract number of $GPGSV lines
2730 novrecs% = satlist$(1,8)
2740 :
2750 REMark Get rest of $GPGSV lines
2760 REMark FOR j= 2 to novrecs%
2770 REMark  satlist$(j)= gpsdata$("GPGSV")
2780 REMark END FOR
2790 REMark Had trouble with that so .. assumed
     always two ...
2800 satlist$(2)= rawdata$ (3)
2810 :
2820 REMark Get RMC data
2830 REMark Too late now, but should use a copy of
     raw data!
2840 gpsrmc$= gpsdata$ ( "$GPRMC")
2850 :
2860 REMark Start at first $GPGSV line
2870 t$= satlist$(1): REMark To leave full data for
     later
2880 f$= field$(chop$(t$,3))
2890 REMark IF NumChk(f$)<>0 THEN NEXT orbits
2900 novsats%= f$
2910 :
```

```
2920 REMark Now have all the data needed for
     display
2930 :
2940 REMark Extract satellites used into satsusd
     array
2950 REMark from $GPGGA line
2960 v$= chop$(gpsAdata$,2)
2970 i= 0: REMark Count for sused REPeat loop
2980 PRINT \" Sats used   : ";
2990 :
3000 REMark Extract time and validity
3010 UTC$= GPSTime$(rawdata$(4))
3020 Inv%= 0: REMark to record invalid time and
     data
3030 IF gpsrmc$(18)= "V": Inv%=1
3040 :
3050 REPeat sused
3060   v$= chop$(gpsAdata$,1)
3070   REMark Mark end of list with -1
3080   IF v$(1)="," THEN satsusd(i)= -1: EXIT sused
3090   f$= field$(v$)
3100   IF LEN(f$) =0 THEN NEXT orbits
3110   satsusd(i)=f$
3120   IF satsusd(i)<10 THEN PRINT "0";
3130   PRINT satsusd(i);" ";
3140   i= i+1
3150 END REPeat sused
3160 :
3170 REMark Spaces at end of 'used' line for
     shorter overwrite
3180 PRINT "      "
3190 :
3200 PRINT " Sats in view: ";
3210 :
3220 REMark Extract data from $GPGSV lines
3230 v$= satlist$(1)
3240 v$= chop$(v$, 2)
3250 FOR i= 1 TO novsats%
3260   v$= chop$(v$, 2)
3270   id$= field$( v$ ): REMark Satellite
       Identifier
3280   IF LEN(id$) =0 THEN NEXT orbits
3290   id%= id$
3300   IF id%<10 THEN PRINT "0";
3310   PRINT id%!;
3320   v$= chop$ ( v$, 1 ): REMark Satellite
       elevation
3330   el$= field$( v$ )
3340   IF LEN(el$) =0 THEN NEXT orbits
3350   el%= el$
3360   v$= chop$ ( v$, 1 ): REMark Satellite
       azimuth
3370   az$= field$( v$ )
3380   IF LEN(el$) =0 THEN NEXT orbits
3390   IF NumChk(az$)<>0 THEN NEXT orbits
3400   az%= az$
3410   j= 0: tint%= seentint%
3420   REPeat chkid
3430     IF satsusd(j)= -1 THEN EXIT chkid
3440     IF satsusd(j)= id% THEN
3450       tint%= usdtint%
3460     END IF
3470     j= j+1
3480   END REPeat chkid
3490   :
3500   IF ShowTrack% =0 THEN
3510     REMark At last can plot satellite in
         position
3520     REMark Plot omly valid data
3530     IF Inv%=0 THEN : spot el%, az%, id%, tint%
3540   END IF
3550   REMark Change data line after 4 sats
3560   IF (i MOD 4)= 0 THEN v$= chop$(satlist$(2)
       ,2)
3570 END FOR i
3580 :
3590 REMark Spaces at end of 'ids' line for shorter
     overwrite
3600 PRINT "      "
3610 :
3620 REMark Print validity
3630 PRINT \"   ";
3640 IF Inv%=1: PRINT "Inv";: ELSE PRINT "V";
3650 PRINT "alid Position  "
3660 REMark Spaces to overwrite longer 'Invalid'
     message
3670 :
3680 REMark Extract Lat and Long
3690 REMark Chopped gpsrmc$ also used by Track code
3700 REMark so use a variable to synchronise
3710 rmchop%= 3
3720 gpsrmc$= chop$(gpsrmc$, rmchop%)
3730 CSIZE 3,1
3740 AT 3,0
3750 REMark Latitude
3760 PRINT "  ";gpsrmc$( 1 TO 2); CHR$(186);
3770 PRINT " ";gpsrmc$(3 TO 10);"'"!gpsrmc$(12);
3780 LatDeg$= gpsrmc$( 1 TO 2)
3790 LatMin$= gpsrmc$(3 TO 10)
3800 Lat= DecDeg(LatDeg$,LatMin$)
3810 IF gpsrmc$(12) =="S" THEN Lat= -Lat
3820 REMark Validity
3830 PRINT "  "; Validity$
3840 REMark Longitude
3850 PRINT " ";gpsrmc$(14 TO 16); CHR$(186);
3860 PRINT " ";gpsrmc$(17 TO 24);"'"! gpsrmc$(26)
3870 LonDeg$= gpsrmc$(14 TO 16)
3880 LonMin$= gpsrmc$(17 TO 24)
3890 Lon= DecDeg(LonDeg$,LonMin$)
3900 IF gpsrmc$(26)== "W" THEN Lon= -Lon
3910 :
3920 REMark Display UTC, extracted way back
3930 AT 6,2
3940 PRINT UTC$;
3950 :
3960 REMark Extract Track data,
3970 REMark    gprsmc$ already chopped for Lat,
     Long
3980 AT 8,2
3990 trak$= gpsrmc$: trak$= chop$(trak$, 7-rmchop%)
4000 speed$= trak$(1 TO 5)
4010 IF NumChk(speed$)<>0 THEN CSIZE 0,0: NEXT
     orbits
4020 Speed= speed$* 1.150779: REMark Convert Knots
     to m.p.h.
4030 PRINT "Track"!trak$(7 TO 11);CHR$(186): REMark
     Bearing
4040 PRINT_USING "     at ##.# m.p.h", Speed
4050 Brg= trak$(7 TO 11)
4060 :
4070 CSIZE 0,0
4080 :
4090 IF ShowTrack% THEN
4100   REMark Show track as a line: colour shows
       speed
4110   INK #dc%,Spink%(Speed)
4120   IF FirstPt%==0 THEN
4130     POINT #dc%, Lon*asprat*Mctr, Lat
4140     FirstPt%= 1
4150    ELSE
4160     LINE #dc% TO Lon*asprat*Mctr, Lat
4170   END IF
```

```
4180 :
4190   REMark Show Speed and bearing as a line, ..
4200   REMark .. length and colour for speed, ..
4210   REMark .. bearing as direction of line.
4220   CLS #dt%: INK #dt%,Spink%(Speed)
4230   POINT #dt%,0,0
4240   PENDOWN #dt%
4250   TURNTO #dt%,90-Brg: MOVE #dt%,Speed
4260   PENUP #dt%
4270 END IF
4280 :
4290 REMark Pause to slow down rate of refreshment
     of display
4300 PAUSE 50*delay
4310 :
4320 REMark End of repeat loop for continous
     display
4330 END REPeat orbits
4340 CLOSE #cs%: IF dtof <>0 THEN CLOSE #dc%
4350 STOP
4360 :
4370 REMark End of Main Program ==================
4380 :
4390 DEFine PROCedure radials
4400   REMark Draws the celestial meridians at 30
       deg intervals
4410   REMark as a series of dots to avoid too dark
       a line
4420   LOCal i,j,interval
4430   interval = 2E-2
4440   FOR i = 0 TO 350 STEP 10
4450     FOR j= 2 TO 100
4460       REMark uses j*interval as a radial
           distance ..
4470       REMark .. which must be converted to x,y
           for plot
4480       IF j*interval > 1 THEN EXIT j
4490       POINT#dc%,j*interval*COS(i*PI/180)
           *asprat,j*interval*SIN(i*PI/180)
4500     END FOR j
4510   END FOR i
4520 END DEFine
4530 :
4540 DEFine PROCedure spot(elvn,azm,id,tint)
4550 REMark Draw a blob in tint at elvn,azm and
     show id
4560 REMark For a polar plot need to convert to x,y
     coords
4570 REMark Code for line plot added later
4580 LOCal x,y,srad
4590 REMark ignore data if id outside possible
     range ..
4600 REMark .. it has happened, usually during
     start up of RX
4610 IF id > maxid THEN RETurn
4620 srad= (90-elvn)/90: REMark zero to one on plot
4630 x= srad*SIN(azm*PI/180)*asprat
4640 y= srad*COS(azm*PI/180)
4650 REMark Copy previous posn as 'old'
4660 posns(id,0)= posns(id,2)
4670 posns(id,1)= posns(id,3)
4680 REMark Save new posn for 'old' next time
4690 posns(id,2)= x
4700 posns(id,3)= y
4710 INK #dc%,tint: STRIP #dc%,tint
4720 REMark Draw sat as blob if first time plotted
4730 IF posns(id,4)< 1 OR Blobs% =1 THEN
4740   posns(id,4)= 1: REMark Remember as blobbed
4750   REMark draw a blob
4760   FILL#dc%,1
4770    ELLIPSE#dc%,x,y,6E-2,asprat,0
4780   FILL#dc%,0
4790   REMark Show sat id in contrasing ink
4800   IF tint= seentint% THEN INK #dc%,0: ELSE INK
       #dc%,7
4810   CURSOR#dc%,x,y,-6,-5
4820   IF id<10 THEN PRINT#dc%,0;: REMark Add
       leading zero?
4830   PRINT#dc%,id: REMark At last, print sat id
4840 ELSE
4850 REMark ... otherwise draw a line
4860   INK #dc%,tint: LINE #dc%,posns(id,0),
       posns(id,1) TO posns(id,2),posns(id,3): INK
       #dc%,0
4870 END IF
4880 END DEFine
4890 :
4900 DEFine FuNction gpsdata$( id$ )
4910   REMark Waits for and reads a sentence of
       data starting
4920   REMark with the string id$ (not sat id this
       time)
4930   REMark Give up if there's no data to serial
       port
4940   FOR i= 1 TO 50
4950     IF EOF(#cs%) THEN
4960       AT 20,3: PRINT "Lines: ";lines!;
4970                PRINT " Good lines: ";GoodLines
4980                INPUT "   Press <enter> to
                finish: ";t$
4990       CLS: CLS#0: CLS#2: CLOSE: STOP
5000     END IF
5010     INPUT #cs%,t$
5020     REMark More checks for dodgy data
5030     IF LEN(t$) = 0 THEN NEXT i
5040     IF t$(1) <> "$" THEN NEXT i
5050     IF t$(1 TO 6) = id$ THEN RETurn t$:
5060   END FOR
5070   IF i > 10 THEN PRINT " No GPS data": STOP
5080 END DEFine
5090 :
5100 DEFine FuNction field$(str$)
5110   REMark Extracts the first field from the
       NMEA message data
5120   REMark after it has been chopped to the
       start of the
5130   REMark field, with comma separated fields,
       so making
5140   REMark no assumption about the field length.
5150   LOCal k%
5160   REMark last field terminated by * at start
       of checksum
5170   REMark so need to check for ',' — if none
       then '*'
5180   k%= "," INSTR str$
5190   IF k% =0 THEN k%= "*" INSTR str$
5200   RETurn str$ ( TO (k%-1))
5210   REMark NOTE errors such as empy string must
       be
5220   REMark dealt with on return from call
5230 END DEFine
5240 :
5250 DEFine FuNction NumChk(a$)
5260 REMark Check if a$ contains a decimal number,
     fixed point
5270 REMark Return: 0= ok; 1= empty; 2= >1 d.ps; 3=
     non-numeric
5280  LOCal i%,a%,NoDecPts%: NoDecPts%= 0
5290  REMark NoDecPts% holds the nunber of dec pts
      found
5300  IF LEN(a$) = 0 THEN RETurn 1
5310  FOR i% = 1 TO LEN(a$)
```

```
5320    a%= CODE (a$(i%))
5330    REMark 48 and 57 are codes for '0' and '9'
5340    REMark .. I just like < more than <=
5350    IF a% > 47 AND a < 58 THEN
5360      NEXT i%
5370     ELSE
5380      REMark Check for dec pt
5390      IF a%=46 THEN
5400        NoDecPts%= NoDecPts%+1: REMark Allow
            one dec. pt.
5410        IF NoDecPts%> 1 THEN RETurn 2: ELSE
            NEXT i%
5420      END IF
5430     RETurn 3
5440    END IF
5450  END FOR i%
5460  RETurn 0
5470 END DEFine
5480 :
5490 DEFine FuNction chop$(str$, skip)
5500   REMark Chops off skip fields from the start
       of str$
5510   REMark Haven't had to chop as far as the '*'
       yet
5520   LOCal i,j
5530   IF ( skip < 1 ) THEN RETurn str$
5540   FOR i= 1 TO skip
5550     j= "," INSTR str$
5560     str$ = str$ ( j+1 TO )
5570   END FOR
5580   RETurn str$
5590 END DEFine
5600 :
5610 DEFine PROCedure DisLine(c%,t$,l%)
5620   REMark Diplays line padded to l% chars with
       spaces ..
5630   REMark .. in channel #c%, the spaces are
       needed
5640   REMark when short lines follow long
5650   REMark Had no success with CLS so far
5660   PRINT #c%;t$;
5670   tl%= LEN(t$)
5680   IF tl%< l% THEN
5690     PRINT #c%;FILL$(" ",l%-tl%)
5700    ELSE PRINT #c%
5710   END IF
5720 END DEFine
5730 :
5740 DEFine FuNction GPSTime$(t$)
5750  REMark Extracts time from t$ -- copy of RMC
      input
5760  REMark Should have used chop$ etc, but by
      this time the
5770  REMark data format seemed stable enough and
      CBB took over.
5780   t$= chop$(t$,1): REMark Remove '$GPRMC'
5790   RETurn '@ '&t$(1 TO 2)&':'&t$(3 TO 4)&':'
       &t$(5 TO 6)&'UTC'
5800 END DEFine GPSTime$
5810 :
5820 DEFine FuNction CheckSum(a$)
5830 LOCal i,aa$,ChkSum,ChkCode$
5840 IF LEN(a$)=0 THEN RETurn -2
5850 FOR i= 1 TO 256: REMark Allow for sentence ..
5860   REMark        .. not terminated properly,
       i.e. no '*'
5870   aa$= a$(i)
5880   SELect ON CODE(aa$)
5890       REMark set ChkSum to zeros at start of
           sentence
5900     = CODE('$'): ChkSum= 0
```

```
5910     = CODE('*'):
5920       ChkCode$= a$((i+1) TO (i+2))
5930       EXIT i
5940     = REMAINDER :
5950       REMark ^^ is 'bit-wise exclusive OR' Jan
           Jones p 40
5960       ChkSum= CODE(aa$) ^^ ChkSum
5970   END SELect
5980 END FOR i
5990 REMark Checksum in data is two hex chars (8
     bits)
6000 IF HEX$(ChkSum,8) = ChkCode$ THEN
6010   RETurn 0: REMark Good result
6020  ELSE
6030   REMark Bad result
6040    PRINT #de%,a$;" ChkSum: ";HEX$(ChkSum,8);"
        Check code: ";ChkCode$
6050    RETurn -1
6060 END IF
6070 END DEFine
6080 :
6090 REMark Follow a series of definitons of charts
     for tracks
6100 REMark Maxlon necessary for lat, lon grid
6110 DEFine PROCedure ChiCityMap
6120   Minlat= 50+49/60
6130   Maxlat= 50+51/60
6140   Minlon= -47/60
6150   Maxlon= -43/60
6160 END DEFine
6170 :
6180 DEFine PROCedure PfdMap
6190   Minlat= 50.7667
6200   Maxlat= 51.05
6210   Minlon= -1-3/60
6220   Maxlon= -.6
6230 END DEFine
6240 :
6250 DEFine FuNction DecDeg(D$,M$)
6260  IF LEN(D$)==0 OR LEN(M$)==0 THEN RETurn 361
6270  IF ',' INSTR(D$&M$) THEN RETurn 362
6280  RETurn D$+(M$/60)
6290 END DEFine
6300 :
6310 DEFine FuNction Spink%(S)
6320   REMark Returns a colour according to the
Speed S
6330   SELect ON S
6340     ON S=0  TO  9.999: RETurn 0  :REMark Black
6350     ON S=10 TO 19.999: RETurn 59 :REMark Brown
6360     ON S=20 TO 29.999: RETurn 2  :REMark Red
6370     ON S=30 TO 39.999: RETurn 236:REMark Yellow
6380     ON S=40 TO 49.999: RETurn 22 :REMark Orange
6390     ON S=50 TO 59.999: RETurn 3  :REMark Green
6400     ON S=60 TO 69.999: RETurn 25 :REMark Blue
6410     ON S=70 TO 79.999: RETurn 26 :REMark Violet
6420     REMark Shocking pink for over 80m.p.h.
6430     ON S= REMAINDER : RETurn 112
6440   END SELect
6450 END DEFine
6460 :
6470 DEFine PROCedure LatLonGrid
6480 REMark For track, prints a grid of lats and
     longs ..
6490 REMark .. each at one minute of arc intervals
6500 REMark parameters set in main program
6510 REMark No need for precise match to window as
     SB
6520 REMark just doesn't draw outside it
6530 LOCal Glat, Glon, Gld, Glm
6540 INK #dc%,13: REMark nice pale grey
```

```
6550 :
6560 REMark Start with latitudes
6570 Gl= Minlat
6580 REMark Need to convert to decimal degrees
6590 Gl= INT(Gl)+(INT((Gl-INT(Gl))*60))/60
6600 Gminl= Minlon*asprat*Mctr: Gmaxl= Maxlon*
     asprat*Mctr
6610 REPeat LatLines
6620    IF Minlat<=Gl AND Maxlat > Gl THEN
6630      LINE #dc%, Gminl,Gl TO Gmaxl,Gl
6640    END IF
6650    Gl= Gl+1/60
6660    IF Gl> Maxlat THEN EXIT LatLines
6670 END REPeat
6680 :
6690 REMark next meridians
6700 Gl= Minlon
6710 Gl= INT(Gl)+(INT((Gl-INT(Gl))*60))/60
6720 Gminl= Minlon: Gmaxl= Maxlon
6730 REPeat LonLines
6740    IF Minlon <= Gl AND Maxlon > Gl THEN
6750      Gp= Gl*asprat*Mctr
6760      LINE #dc%, Gp,Minlat TO Gp,Maxlat
6770    END IF
6780    Gl= Gl+1/60
6790    IF Gl> Maxlon THEN EXIT LonLines
6800 END REPeat
6810 END DEFine
6820 :
6830 REMark My programs always have this procedure
     at the end
6840 DEFine PROCedure backup
6850   SAVE dos1_GPSTalk_QLMags_Figure11.txt
6860 END DEFine
```

'Ideally the program would be a paradigm of nice structuring, with all functions and procedures following the main part. What I've got is the messy result of developing it in stages as I discovered what could be done, first to show orbits, then putting 'DEF PROC' round that part, keeping the common channel definitions, display of satellite ids, lat, lon, time etc. and then adding the code to display the track as an alternative, with yet more additions to get line plots of orbits. Whenever I try to tidy it up (e.g. I just realised I could usefully use CLS in some places) I introduce more errors, so I present it as it is -- IWSWFI.

I hope that the REMarks, and the pictures of the resulting displays, will make it all clear.

In simulation mode the program always takes its data from the file win1_gps_sats_data, and correspondingly stores data in the same file, so I keep backup copies of the original data files and copy the one I want to display. I use variables, Laptop%, ShowTrack%, Blobs, sim and dtof, to pick out what it is to do, using ED to change them. The REmarks in the early lines of the program explain what they are for, I could put in queries, INPUTs, validity tests, and defaults, but that comes under CBB; sorry.

There were a lot of REMark lines left over from error tracing, I keep needing them still, but I've deleted them from figure 11 to keep it a bit shorter.

Orbits

I wanted a circular sky picture, but the raw x and y values gave an ellipse. I use "asprat" (aspect ratio) to adjust the x values to get circles, with the value fixed by experiment until it looked right. I tried "G_RATIO 1" but that too gave a slight ellipse, most likely because I don't know how to use it properly. You may, of course, have to adjust asprat for your display.

For the orbits, the view is as looking down on the satellites from outer space, a view looking up at the sky would have East and West reversed: I found the former more intuitive, so I used that. Satellites are initially drawn as blobs with their identifier numbers, dark blue background for those used for the fix, light blue otherwise. I expected to have to do too much programming to replace the underlying grid when the satellite moved, so the first version simply keeps overwriting with the blob, producing a long 'sausage' along the orbit. You can tell the direction the satellites travel because the number is at the last position plotted. Figure 5 shows the result of a five hour recording at home, with a two minute delay between recordings. Figure 8 is the same plot as figure 5 but interrupted part way through to make clearer where the orbits are going. A lot more than eight satellites appear overall, but note that most of them rise or set during the period.

Figure 7 shows the whole screen for some orbit data, with the other windows used by the program: to the right is #dc%, the instantaneous display of some of the data; above this is window #de% used in development for error and other messages, including lines where the checksum test fails; and below is #dd% which shows the current five sentences from the GPS receiver.

Later I tried a single starting blob with a thinner line drawn from then on, but I don't really like that, figure 8, but I think it too helps to make sense of figure 5. I suppose sprites could solve it, i.e. to get a line always ending in a blob at the satellite's current position, but that comes under CBB (so far) again.

[To view figures 5 to 10, please see issue 2 of QL Today Volume 11 which contains part one of this article - Editor]

Track over the ground

For the track display as in figure 9, either on the move, or back in the shack from the recorded data, the window represents a map on which the track is drawn, a direct plot of latitude and longitude for each point. I set Minlat, Maxlat, Minlon and Maxlon, taken from an OS map of the area, to define the edges in decimal degrees (S and W are negative), and used the difference between Maxlat and Minlat to set the SCALE. The E-W width of the map is then set by the window definition; the SW corner by (Minlon,Minlat), also in SCALE; and Maxlon is really used only for the limits of the grid of latitude and longitude lines. Since SB does not complain about off-window drawing, and the map is drawn only once, I left it like that. I define the four variables in a number of PROCedures to choose the area and detail I want to see, again using REMarks and ED to select the one I want. More CBB against making this easier. I've removed most of the maps in this version to save space.

Figure 9 shows a trip from Bognor Regis to Petersfield via the A259, the A286 the B2141 and the B2146. If you want to compare it with a real map, it's all on OS Landranger Sheet 197.

To get a plot that would correspond to the OS map I had the same problem as any cartographer representing the Earth's surface on a flat map. The compromise is to adjust the longitude by the cosine of the average latitude, which works well over a small area. Mercator charts (which includes OS maps) use this method hence I call the adjuster Mctr. To be honest I just kept fiddling with it until the plot looked right when compared to the OS map. A 'To Do' is to look into displaying an image of a proper map and drawing the track on it, it should be fairly straightforward if I photograph a map with a digital camera and could convert it to a format that QPC will display. I would note the latitudes and longitudes of the corners of the map and adjust the SCALE and origin of the GPS display to fit.

A small window shows the instantaneous speed and direction as a line, length proportional to speed, in the current direction, with North at the top. Turtle graphics made this simple to program, but didn't allow the correction for aspect ratio, so it's a slightly distorted picture, but CBB came in handy again.

I've coloured the track, and the small pointer, to show the speed, based on the standard resistor colour code: black= 0 to 10, brown= 10 to 20, red= 20 to 30, orange= 30 to 40, yellow= 40 to 50, green= 50 to 60, blue= 60 to 70, violet= 70 to 80 and then a non-standard shocking pink above 80. I have found that the speed given in the $GPRMC line is sometimes way out, over 70 m.p.h. on a road where I know we didn't exceed 50. That was when we were under a canopy of trees, and the data from around that time is really dodgy. This is something else to look into.

No static, black and white, figure can show the dynamic effect of running the program, with colour, and, believe me, it is much more exciting to watch the real thing: to me, well worth the trouble.

Processing the data

I soon found that the early versions of the program collapsed due to corrupted data from the receiver. However, going through the data as it arrived looking for the corrupted lines became too long-winded, now I read in all the five sentences in one go, so that processing time doesn't mean that some data comes from the next second's lot, Ignoring the whole set of five lines if an error is found is maybe a bit OTT, but CBB ruled here too. As it stands, I catch only every other second's data, and the main display of track data does not always correspond to the raw data shown at the same time, which is a puzzle. Looking into this is another 'To Do'. (Well, since writing that I've looked through the program prior to sending it off, and realise that the answer is that I don't always use the copy of the data -- too late to do it now, so still a To Do.)

DIY

If you want to try this out, you don't, of course, need to do all that I have -- I was exploring into the unknown. There's no need for the Icebreaker and LCD display, no need either for the PPS and ALMRDY, but I love flashing lights: you could go straight to QPC, but I would keep the op-amp buffer(s), to protect the GPS module; and the MAX232 might be necessary too. A GPS module with a less fiddly interface might be a good idea, unless you're into fine p.c.b. etching.

More To Do

Apart from the things I've already mentioned, I might change the baud rate to 9600 and get processing done within a second so I can collect all the data. I could try out saving only the necessary data instead of the whole set of sentences, to keep the file sizes down, although studying the details of recorded data has been very instructive.

I've already developed a SB program to make an average of the positions from the five hour data file, it will be interesting to repeat this with the antenna moved a few metres and see if the difference in position can be measured by this method.
There's still the prospect of making a hand held device based on PICs, but that would take it outside the QL area.
I'll report here any interesting developments, and would like to hear from anybody who has any queries, comments, suggestions or corrections, via email: harryweston·beeb.net (it's a long story). Please make it clear in the first few characters of the subject field that it relates to QL matters; I get a lot of spam, and delete anything I don't recognise, without downloading or reading it.

Acknowledgements

I must thank many people: Roy Wood for giving me the opportunity to go on about GPS at Hove, and for providing the projector and improvising a screen; Tony Firshman for helping me with an unfamiliar Laptop computer, and his persistence that got the GPS antenna outside and made a demonstration possible; Geoff Wicks and John Mason for encouraging me to produce this article - I have enjoyed working on it and learned a lot more about GPS and the QL by having to write it all out. Finally all those others who work so hard to keep the QL community alive, for little profit, but a great deal of benefit to the rest of us.

Sources and References

I have no connection with any of the organizations cited except as a happy user of their products.

(1) **http://www.rfsolutions.co.uk/acatalog/Board_Level_GPS_Receiver_M**
Data sheet DS301

(2) *--do-- data sheets DS031 for GM001 and DS303 for the antenna*

(3) **http://www.holux.com.tw Module GM210**

(4) **http://www.ordnancesurvey.co.uk/gps**
A mine of information. In particular the OS document "A guide to Coordinate Systems in Great Britain", is essential: it's one of the most useful and most quoted. The site also has details of the Active and Passive networks of reference sites for Differential GPS.

(5) **http://www.english-heritage.org.uk/upload/pdf/where_on_earth_are_we.pdf**
A booklet "Where on Earth are We? The Global Positioning System(GPS) in archaeological field survey" A very readable and beautifully presented paper. I recommend this as the first to read. Only one caveat: the diagram to illustrate Differential GPS on page 9 is misleading as it shows the paths from the satellites diverging towards the two receivers, whereas the satellites are so remote that the paths are almost exactly parallel, which is the whole point of it.

(6) **http://www.wgs84.com/**

(7) **http://www.gps.gov.uk/** *the gateway to a vast range of useful information.*

(8) *"Greewich Time and the Longitude" by Derek Howse ISBN 0-85667-468-0 p160 .*

(9) *NMEA itself can be found at* **http://www.nmea.org,** *but it seems that you have to be a member to access their information. A useful guide to the standard sentences is accessible via* **http://www.gpsinformation.org/dale**

(10) **www.colorado.edu/geography/gcraft/notes/gps/gps.html**
Peter H. Dana, The Geographer's Craft Project, Department of Geography, The University of Colorado at Boulder. A detailed but rather technical overview of the GPS system by, I believe, one of the designers.

(11) **http://sideshow.jpl.nasa.gov/mbh/series.html**
This internet address is on a map of the world showing the relative tectonic movements. I have lost the original reference.

(12) *"Everyday Practical Electronics" magazine:* **http://www.epemag.co.uk**

(13) *SMSQ/E for QPC manual revision 2.08 pages 9, 10*

(14) *SMSQ manual revision 8 pages 27, 28*

(15) *"QL SUPERBASIC the definitive handbook" by Jan Jones ISBN 0-07-084784-3*

(16) *"Eats, Shoots & Leaves" by Lynne Truss ISBN 1-86197-612-7 The source and inspiration for all those colons and semi-colons.*

Small Ads

WANTED

External disk drive units for the Sinclair QL, empty cases with power supplies will do!
Rich Mellor
3 Dale View Court,
Fulford
Stoke-On-Trent
Staffordshire
ST11 9BA
United Kingdom

Dear Readers, please remember: small ads are free! They need to be QL-related, but they don't cost you anything. Just send them to us - whether you sell or search something.

Code Timer - Benchmarks

by Stephen Poole

Back in the early eighties I was using my QL a lot in my surveying job, which required thousands of trigonometrical calculations to be done, to convert theodolite readings into vectors, susceptible to allow the drawing of plans. So my programs were scientific number-crunchers, very demanding in processing power. One day I wondered how SuperBasic maths functions compared to their machine code equivalents for speed, and decided to write a relative BenchMarking routine, which would allow me to compare SB operators among themselves.
First appeared the problem of system overheads: QDOS SuperBasic was interpreted line by line, so my bench-mark routine had to be a one-liner, and typed into #2, on line 1 at that! Every detail had to be considered. Even names had to be just one character long! But to set up my routine, I still had to measure overheads, so I started the code by RUN 2, which measured the overheads and which then jumped back to the loop which would execute the code I wanted to test on line 1...
This allowed me to compare code fragments appended to the line 1 loop, but I still had to make sure tests were compatible: to time `x=PI`, for example, I first had to time `x=3.142` as it would be totally meaningless to compare it to an empty loop. And the results began to become interesting. For example, `PRINT 'QL '` would be different to `PRINT 'QL'!` or `PRINT 'QL ';` or `PRINT 'QL '&chr$(10)` or `PRINT QL$...`
Indeed it became clear that QDOS exhibited strange variations in the execution speeds of all manner of operations, which depended uniquely on their internal machine-code efficiency. In view of these observations, I entered into correspondence with John Southern, who was at the time Quanta Magazine Editor, and sent him my results as soon as I found anomalies. I bought a 68008 processor reference book, and studied the speeds of the pure machine code performance of operators, relative to one another. In this way I was able to make observations about SuperBasic operators relative to processor operators, even though I could not take measures directly.
From these observations, it appeared that Superbasic was running slow in divisions, multiplications, certain PRINT operations, POINT, LINE and other graphics functions, and a long list which I do not have space to mention here.

It was clear that SuperBasic was not fully optimised under QDOS. Then I bought TURBO, which contained code to 'Profile' parts of programs, that is, to measure the speeds of their component parts, so as to optimise thereafter their inner loops, under Turbo Basic or Machine code.
Later still, SMSQ/E arrived, which had been extensively rewritten, although based on QDOS, and was 12.5 times faster on heavy-duty number-crunching graphics programs. That is to say, an acceleration equal in speed to compiled TURBO basic! Since then I barely use Turbo, although it does possess a very useful Toolkit. But unlike Turbo-Compiled code, SBasic integer-loops are not faster than floating-point ones. In fact they are only 90% of their speed. For some reason I never timed operators under SMSQ/E: I suppose it was because it was so much faster that I was flabbergasted!
Recently I was sifting through my stock of old programs when I fell upon the QDOS Code-Timer. I tried running it on QPC2, but it was clear that with PC overheads the routine (written for the multi-tasking QL), was incompatible. So I set about rewriting it for QPC2, and this proved much easier, as SMSQ/E is fully compiled when RUN, so it does not need to be on the first code-line! The result is the CODE_TIMER_bas program, which you must tweak to adapt to your machine specifications. To do this, just run the program, and it will print a list of figures. Note the largest negative figure, and EDIT it into the ‹overhead=figure› statement in place of ‹overhead=0›. Now your system overhead is set and we are ready to start benchmarking.
Computer Magazines often allow you to download 'benchmark' programs to test your machines. Way back in the seventies, people used the method of the 'Gibson Mix'. This was a selection of operations generally used by scientific number-crunching programs, which uniquely measured processor perfomance. This mix has has been superceded by simply running the very latest 3D-perspective adventure games flat-out with maximum frame-rate, surround-sound and much ker-splatting of graphics! Otherwise you are more likely to use 'Whetstones' or 'Dhrystones' which test the sort of operations needed by your average user: multiple Hard-disk acces-

ses, memory read-writes, sorting large databases, using two programs at once, etc. But I wanted a routine like my QDOS one that would test the individual operators of SMSQ/E, more particularly on QPC2.
So here are some of the results from my initial trials:
x=y 40404 cycles, x=(y) 39668, x=y% 85613 , x=y$ 73899, x=3 47406, x='3' 78743, x=z(1) 23395, x=z(i) 74294, x=z%(1) 73419, x=z$(1) 61576, x=z(n(i),n(i)) 142063, x=z(1,2,3) 129693. As can be seen by examining the listing, the higher the number of cycles, the faster the operation, as what is being measured is the number of loops executed per second, (minus the overhead).
Now, z=x+y 114887, z=x-y 114447, z=x*y 114363, z=x/y 114877. Here the arithmetic operators are all equivalent in speed, and run much faster than the above simple assignments.
y=SIN(x) 151448, y=COS(x) 114447, y=TAN(x) 151496 y=ASIN(x) 134581, y=ACOS(x) 133472, y=ATAN(x) 144927.
Here again FuNctions RETurn results even faster than simple arithmetic operations! x=PI^.5 135550 , x=SQRT(PI) 147222, x=SQRT(3.142) 96911
Now using the floating-point constant returned by PI as an actual parameter is even faster still!
In General: Integer and string handling on SMSQ/E are respectively 39% and 86% of the speed of floating-point operations! As far as assignments are concerned, it is invariably faster to assign variables from floating-point array structures. It is faster still to assign constants from within FuNctions! Two-dimensional arrays indexed with function-constants are the fastest of all!
Incidentally, there would appear to be a bug with assignments of the type x$=3 , where speed can oscillate from simple to double, (e.g: 10 seconds for 87925 or 153338 cycles)! All the maths operators seem to be fully optimised and produce almost the same timings for + - / * on eqivalent data. All the various graphics functions too, seem to run flat out at similar speeds, whether using BLOCK, POINT, CLS, WINDOW TURTLE CURSOR or PRINT ' '. There would appear to be a very slight advantage in using FOR loops compared to REPeat, but the difference is negligeable.
To sum up, it is matching internal coercion to data types which offers the most scope for program optimisation. The machine has dozens of different permutations of assignment types to juggle with. The fastest assignments I have obtained are, surprisingly, of the type:
x_var=float_array(FuNction(constant_float), function(constant- _float)).
Compare this to x=y: eg: 142063 cycles for the former and 40404 for the latter, a speed of just 28.4% for an 'ordinary' assignement! This has important repercussions on program design. Under QDOS, calling procedures and functions always slugged the QL. (It was faster to use GOSUB). Now use them to your heart's content. They will improve your programs considerably!
QDOS was very massively improved by Tony Tebby to produce SMSQ/E. But for the time being it is up to you to carefully choose assignment-types in your own programs. The scope for program optimisation is not trivial. A threefold increase, (or at least a doubling) within inner loops can be expected... If so requested, I will find time to publish tables of all SMSQ/E operations, to help you make your choices. In the meantime try experimenting on your own machine. But do make sure that the statements you are comparing are equivalent. Perhaps someone will try to optimise internal coercions on the QL as the next Quantum Leap in accelerating the system?

```
100 ::
110 REMark Code_timer_bas by S.Poole,
    v20feb2007
120  REMark First run and tweak
     overhead=ABS(result),
130  REMark using the largest negative
     result printed.
140  CLEAR : OPEN#1,con_128: CLS
150 :
160 FOR main=1 TO 10
170     overhead=ABS(0)
180     REMark Wait for the clock to
        tick:
190     d=DATE: REPeat loop: IF
        d<>DATE: d=DATE: EXIT loop
200     :
210     FOR cycles=0 TO 512
220         :
230         REMark no overflows please:
240         FOR count=0 TO 32767
250             ::
260             REMark Put code to be
                tested here!
270             ::
280             IF d<>DATE: EXIT cycles
290         END FOR count
300     END FOR cycles
310     PRINT\ overhead-((cycles*32768)
        +count)
320 END FOR main
330 ::
```

Programming Assembler - Part 18

by Norman Dunbar

In the last exciting instalment of the series, I mentioned that I would be looking into the bowels of QDOSMSQ to see if I can find a useful subroutine to convert a string of ASCII characters into a long value in a register. This was suggested by comments from George Gwilt when he mentioned that he was surprised that I didn't have a reusable routine to do this conversion.

As ever, I like to take the lazy approach to writing code. If someone else has done it for me, that's a bonus. Inside QDOSMSQ there is a vectored routine called CN_DTOF which reads a string of characters and converts those to a floating point value on the maths stack. This routine can be entered with D7.L holding the address of the first byte of memory AFTER the final character of the string, or with D7 set to zero.

In the latter case, the CN_DTOF routine simply keeps reading until it comes across any character which is not a valid digit, decimal point or 'e' in the buffer. In the former case, the routine stops when it reaches the address in register D7.L or if it hits an invalid character before then.

On exit, the buffer pointer is pointing at the character after the buffer or at the invalid character, unless an error occurred, in which case A0.L and A1.L are restored to their values on entry.

So far so good, we have a floating point value on the maths stack at 0(A6,A1.L) but we wanted a long value from our routine. This too is easy. Thinking back to the article on using the arithmetic package, we can use the RI_NLINT operation to convert a floating point value down to a long word. Once this is done, it is a simple job to copy it off the maths stack into our data register and we are done.

All conversion 'problems' for the character data have been dealt with by QDOSMSQ as have problems of overflow and so on when we convert from FP to LONG.

How easy can it get?

Obviously, we might have a small problem, after all, isn't the maths stack provided for use by SuperBasic routines only? Well, the code in this article shows that this is not the case, provided a couple of simple rules are followed.

Rule number one is that A1.L has to point at the byte just above the top of the maths stack - at the highest address in other words.

Rule number two is that you must have enough space on the maths stack for the operation(s) to be carried out. It is possible that some routines will need working space on the maths stack. This must be catered for or you may find that the maths operations corrupt data below your maths stack.

According to Dickens, the CN_DTOF vector uses about 30 bytes of space on the maths stack. So, for this conversion routine to work, you should set up a maths stack with at least 30 bytes - although it wouldn't break the system to use a bit more for safety. I'm using 30 long words, which should be ample.

The maths stack, while looking special, has to be considered for what it is, it is just a chunk of memory somewhere in the system.

The following is our conversion routine in all it's glory. As you can see, there is not much to it.

```
;------------------------------------------------------------------------------
; Useful routine to convert a buffer of ASCII into a LONG word in a register.
;
; Entry Registers :
;
; A0.L - Pointer to first character in buffer (not size word).
; A1.L - Pointer to an area of AT LEAST 30 bytes for a maths stack.
;
; Exit Registers :
;
; D0.L - Error code, or zero if no errors. (Z flag set for no errors).
; D1.L - Value of converted ASCII string.
; A0.L - Updated pointer. First character after all valid numerics (and 'e')
;        or first character after end of input in nothing was invalid.
; Rest preserved
;
; Error Exit Registers :
;
; D0.L - Error code, or zero if no errors. (Z flag set for no errors).
; D1.L - unknown.
; A0 - Preserved = pointer of start of buffer on entry.
; Rest preserver.
;------------------------------------------------------------------------
ri_nlint equ       6                ; Code to convert FP to LONG

convert  movem.l   d2/d7/a1-a3,-(a7)  ; Save workers
         suba.l    a6,a0            ; Relativise buffer address
         suba.l    a6,a1            ; And the maths stack
         moveq     #0,d0            ; Assume no errors
         moveq     #0,d1            ; Zero result
         moveq     #0,d7            ; For CN_DTOF
         move.w    cn_dtof,a2       ; Convert ASCII to an FP number
         jsr       (a2)             ; Do conversion
         tst.l     d0               ; OK ?
         bne.s     restore          ; No, bale out.
```

The entry point to our routine is at the 'convert' label above. We start off by saving all the registers that we are going to use, or that will be trashed by the various QDOSMSQ code.
Once that has been done, we subtract the current value of A6 from the two pointer registers as these addresses have to be A6 relative for the maths package code to work.
Next, and the most complicated part of the code is to convert our buffer load of characters into a floating point number on the maths stack. If there were conversion errors then we abandon ship and bale out.
Conversion errors occur when there are illegal characters in the buffer – more than one decimal point, two or more 'e' characters etc. Note however, that conversion will stop when a non-valid (but non-error causing) character is found. So '1024K' will result in the value 1024 being created and then conversion would stop.

```
;------------------------------------------------------------------------
; We now have a floating point value on the maths stack at 0(a6,a1.l).
; Convert that down to a long word.
;------------------------------------------------------------------------

         moveq     #ri_nlint,d0       ; FP to LONG
         moveq     #0,d7              ; For maths package
         move.w    ri_exec,a2         ; Execute one maths operation
         jsr       (a2)               ; Do it.
         tst.l     d0                 ; OK ?
         bne.s     restore            ; No, bale out
```

The second part of the code above, is where we convert the floating point value on the maths stack into a long integer. This uses the aforementioned maths package to do the conversion.
Any errors such as overflow will be trapped and returned in D0. We test for this on return from the RI_EXEC and if we have a problem in conversion, we bale out.

```
;------------------------------------------------------------------------
; We now have a long word on the maths stack 0(a6,a1.l).
;------------------------------------------------------------------------

         move.l    (a6,a1.l),d1       ; This is our value
         adda.l    #4,a1              ; Tidy maths stack pointer

restore  movem.l   (a7)+,d2/d7/a1-a3  ; Restore workers
         adda.l    a6,a0              ; Unrelative the buffer again
         tst.l     d0                 ; Set flags
         rts
```

The above simply copies the long word from the maths stack into the D1 register ready to return it to the caller, tidies up the stack and restores the working registers. We exit with the Z flag set if no errors occurred and unset otherwise.
On exit, the address in A0.L points at the first

character after the string of digits that were converted – in an input buffer, for example, this would be the linefeed.
The QDOSMSQ routines to convert the ASCII into an FP number have 'interesting' register settings on exit. If no errors occurred then we exit with A0.L set to point at the character after the end of the buffer, or, at the invalid character that caused conversion to end. If there was a conversion error, the value in A0.L is reset to that on entry – the pointer to the first character in the buffer.
My code exits with the registers set as described in the code header above.
As a quick example of testing the above, and just to prove that it does work, here is a small test harness. Save the following as a new file names test_asm.

```
test     bra.s     test2

result   ds.l      1                 ; One long word for the result
         ds.b      1                 ; One byte for the terminator

fp       dc.b      '1234567.89x'     ; The fp number in Ascii plus an invalid
                                     ; character
         dc.l      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0  ; 15 Long word for a maths stack
msp      equ       *                 ; This is where A1 needs to point, the STACK
                                     ; TOP

test2    lea       fp,a0             ; Buffer holding Ascii
         lea       msp,a1            ; Top of maths stack
         bsr.s     convert           ; Convert from ascii to long
         lea       result,a1
         move.l    d1,(a1)+          ; Save result
         move.b    (a0),(a1)         ; Terminator
         rts

         in win1_source_convert_asm   ; Load in the utility code
```

Save the file and assemble it. To test it all out, the following is all that is required:

```
ADDR = alchp(1024)
LBYTES win1_source_test_bin,addr
CALL ADDR
PRINT 'Result = '; PEEK_L(ADDR+2)
PRINT 'Terminator = '; CHR$(PEEK(ADDR+6))
```

Which in my case gives me a nice long value of 1234568 for Result and a terminator of 'x'. IN the event of an illegal FP number being converted, say one with two decimal points or two 'e' characters or whatever, an invalid number error will result. If the FP value cannot be converted to a LONG without overflowing, an overflow error will result.
So, there it is, a small piece of code (around 156 bytes in my test_bin file) to convert a string of ASCII characters into a LONG word. How easy was that then?
Now, just this week I have sold my house and so my wife Alison and I are in the process of looking for a new home. This means that I might not have email etc for much longer so I cannot guarantee whether I shall be writing in the next issue or not. Hopefully I will be, but just in case, I apologise in advance for my absence!
See you soon for more exciting code!

Byts of Wood

by Roy Wood

Just after putting the last column to bed I went off to attend the Byfleet QL Workshop. The attendance was rather sparse although the quality of those visiting was high and I did have several good conversations with users, Quanta Committee members and traders. For me, the whole business of going to these shows is more social than commercial - which is just as well given the low number of visitors and subsequent low sales.
This show was the last one they will have at this venue so it was a rather sad event. Some of the people I met at the first of the shows that I have attended over the last 10 years are now no longer with us and I am sure that some of the people who were not there this time only missed out because of bad health or sheer age. It was, however, good to see that some of the younger visitors to

the show were very enthusiastic and positive about their QL involvement.
As always there was talk of how we could improve the attendance and enthusiasm for QL Workshops. To me that whole thing hinges upon the second word, 'Workshops' When the QL was in its heyday many of the people who attended did so to set up their systems and show off the stuff they were doing as well as to buy new hardware and software. With very little of the latter two items on the cards, it was good to see that some of the people who were there did set up systems and show off their current obsessions. I think, if we want to extend and improve the use of QL systems and the numbers or users we need, firstly, to apply ourselves to developing new programs and to making more of a Workshop at the shows. QBranch is always happy to consider new programs so, if you have anything you think others may use, send it to me.

Quanta and the Workshop System

These workshops are not a shows in the sense that you turn up to be entertained - although some of our QL enthusiasts can be quite entertaining, sometimes unintentionally. It should be based on feedback and participation by its members. Some of the QL Users have a vast amount of knowledge and a QL show is one way to tap into that knowledge productively.
There is often a lot of talk about building the shows up to attract people to attend but I think maybe getting people to interact would be a more attractive proposition. You can rely of Tony Firshman being there for most shows to help out with hardware problems and Geoff Wicks is also usually available to discuss software. Even I can offer help and advice so why not make the effort to attend a show and see what you can gain? No purchase necessary, as they say.

No Commercial Potential

It comes as a fitting point, straight after writing that piece, Geoff Wicks announced that Just Words would no longer be a commercial organisation and that all of the software previously marketed by him would become freeware - subject only to the usual costs of copying and supply. It was good to see that this announcement did not mean that he was no longer planning to attend shows and was not planning to close operations completely.
I have often touched on the subject of commercial versus free software in this column and I am afraid that I am going to have to talk about it again in this regard. I have never had any problem with people wanting to make the fruits of their labours free for anyone to use and I have happily used some of it in the past. As you all probably know, I have been fairly scathing about the tendency some authors show towards the excessively anal rulebook. The insistance on including source code along with the program does seem to have relaxed, for some at least, to the state in which, putting the website address where the curious can download the code, will suffice. I don't know if other authors have relaxed the rules about not even charging for copying and return posting & packing. If not they should do so in order to get their efforts to a wider audience.
It is evident that, over the last year at least, it has been the people like Dilwyn Jones who have contributed the most software to the scene. (Hey look Dilwyn, I have given you a mention without a tasteless joke attached!). I do not belittle the efforts and products of the freeware authors at all. Dilwyn's programs last year were very good but I really do not want the QL scene to become a completely free platform. I would like to see some more ambitious projects being tackled and I feel that the thought that, after a few months of hard coding, there might be a small tinkle of loose change heading your way, might tempt people into doing something. No-one expects to get rich here or even cover the time spent out but sometimes, just sometimes, earning enough for a celebratory drink might just be the icing on the cake.

Backup Blues

Linking nicely to that section I found myself looking at backup software over the last few weeks as I wrote the current 'Start Here' section for this magazine. There are four backup utilities available for QDOS/SMSQ systems and I looked at all four while I was writing the piece. Since it was not meant to be a software review I did not go into writing about how they worked and how to use them but I did run them up and look at the way they approached the situation. I realised that we do not have a really good looking, easy to use, backup utility.
All but one of the four are non pointer driven and all four really need a bit of work to get them up to scratch. QBranch currently keeps the Knight Safe 3 on its books but the current version does not really support the high colour SMSQ/E and does lock up under the current version. Mark Knight, the author, is no longer involved (although I am sure I could reach him should I have royalties to pass on). When we last spoke he said he was not interested in doing any more work on it.
Norback, the one PE program in the batch, does work although it has one drawback in that it will crash out of the backup if it hits a file it considers to be corrupt. This can be very frustrating. Apart

from that it is a very good program and has stood the test of time well. Norman Dunbar did tell me that Winback had a 'problem with something in SMSQ/E' but he could not remember what that was now. I do remember using that ages ago and it was a good program too but I did not try it out this time round.
If there is someone out there looking for a good project for the long winter nights then I suggest that this might be it. I don't care if you want to do it as a freebe or as a commercial/shareware option.
I would like to suggest a few areas it should cover:

1. Compressed archive - one of the best features of THe Knight Safe was its ability to produce a compressed set of files keeping the backup small.
2. Restore - it should be able to retrieve one file from and archive if needed.
3. 'Incrementality' - it should be able to find files changed since the last backup and ignore those that have not changed.
4. It should stop when it finds a corrupted file and give the user the option of ignoring that file and continuing the backup - logging the bad file.
5. It should have keep a log of which files it has backed up, which it has ignored and which it considers corrupt.
6. It should be simple to use
7. It should look good

If you think that is not enough to be going on with then how about throwing in a disk defragmentor, utility to compare and flag similar files in different directories and delete corrupt files that the usual range of file managers cannot touch.
Now off you go and write it.

Peter Fox - Clocking Off

One person who can be relied upon to toss a spaniard into the works at a show is Peter Fox - and I mean that in a positive way. He can find problems that no-one else can and he did well at the Byfleet show.
We installed a new battery on his Super Gold Card. Simple procedure,- just remove the old one checking to see where the '+' sign embossed on the battery is and then install the new one making sure that it is

a) Aligned the same way
b) It has no bent legs (only two are required - the other two are for stability)
c) It is firmly in place.

Then re-assemble the unit and boot the system. After that all you have to do is reset the clock and the Auto Boot facility if you need it. Well we did this and after a little struggle because the new battery seemed to have slightly thicker legs than its predecessor, it all went back together. We watched as Peter set the clock - only to find it was twelve hours out. Not a big problem you may think but it would not get set correctly. The Minerva clock - seen at startup, was correct and in 24hr mode but the QL clock, viewed from SMSQ/E stubbornly refused to show the same time.
At first it was just me standing there adding 'useful' comments but we soon attracted a small gathering with John Hall, Per Witte, Phil Jordan and Tony Firshman all staring uselessly at Peter's recalcitrant timepiece. I mean how many QL experts does it take to change a QL clock?
I would like to say that we solved the problem but, by the time we left the building, the clock was still stubbornly twelve hours out. Anyone have any ideas about this?

Perchance Two Screens

In my Start Here article on display setup and in other articles I have mentioned the option which appears in the QPC2 configuration screen and usually reads 'Primary Display Driver'. I had always described the function of this setting to people as selecting which graphics display to use when you have two screens connected to the PC. This was what I believed it did.
At work I use two screens and, in an idle moment the other day, I decided to set QPC2 to appear on the smaller screen to the right. I clicked on this option and all it said was 'Primary Display Driver'. I checked the Windows display options and that definitely had two display adaptors shown which were both configurable. I tried a few different options and could make no sense of it so I emailed Marcel. This was his reply:
'This stems back from the Win95 times, when dual-screen was different from today. These are actually 2 different concepts. Back then you had 2 graphics cards with 2 different drivers which were completely separate. That's what you can select in QPC.
Today Windows manages multiple view screens like a big desktop, so from the application point of view (or rather, Direct X POV), there is only one graphics card present, no matter how many monitors are attached. The option is pretty useless all in all, especially today, but it was a possible choice, so I implemented it.'
I do remember setting up a two monitor system using two different graphics cards in Windows 98 when I ran the shop back in 2000 and I think it was then that I first looked at the Display options in QPC2. So that is that laid to rest then. You can

use QPC2 on either screen on a modern system, of course, just by dragging it there so the option is not really needed any more.

Vista Packed

By the time you read this the latest incarnation of Windows will have made it to the shops and be shipping on many of the PCs that are for sale. It amused me to note that, in the reviews I have read so far much of the excitement has been to do with the cosmetic changes and very little is being said about the actual practicality of the system.
Many of the reviewers have enthused about the new 'Aero Glass' interface in which the windows themselves are 'almost transparent'. I thought that was the definition of a window after all - a space in a solid structure that you can look through. Seems that Bill Gates' team has only just found the dictionary and thought 'oh, that is what it should do'. This is almost as if the inventors of the original windows that were placed in walls back in the mists of time proudly announced their new 'Wall Interface' and filled them with Wood leaving people puzzled why it was not just a wall.
A lot of people will be comparing this to the MAC interface design but, in reality it is all just frippery. Computer interfaces are just fashion accessories after all and the important thing about it is how much difference does it make to the actual processes that you may want to run on the computer itself. This seems down to the age old argument about whether you should use a mouse or not/have icons on the screen or even use more than four colours. It is just individual preference after all.

Safe and Sound

The thing that will be exercising the minds of many people will be the security issue. We have recently seen a change of emphasis in that area. A shift from small time, back room hackers and virus writers producing code for fun or out of a sense of teenage dispepsia just to screw the world (or at least the Windows World) up. These days there is more malevolence at work.
Many hackers are now in the pay of underworld gangs trying to set up an army of 'netbots' and 'spambots' - infected computers that can be used in denial of service attacks and as generators of the vast swarm of spam that swamps the internet. Whereas, in the past, all of this has been directed against the reviled world of Gates and Co, since it is now driven by financial imperatives, other systems are finding that the flaws and holes, which have sat undetected or, at least ignored, in their code are now being exposed and need patching. Will the new Windows be any better than its forbears? Who can say? In the end the biggest security threat is sitting right there with his/her hands on the keyboard and that won't change no matter what system is on the screen.

Hardware and Software Support

We have been running a Beta version of Vista at work for some time but it is only on a spare disk in our test rig. In our experience so far there is a lot of hardware that either doesn't work or is not well provided for. I have not had a chance to try much software on it yet but that does bring me on to my next point.
The first thing I did when I got a few spare moments on Vista was to try QPC2 out. I found, to my surprise, that it did not work. This is the first version of Windows on which whatever version of QPC2 that is current at the time has not 'just worked'. I was about to report this to Marcel when another QL User (Per Witte I think) did so on the users list. Just as I was putting the finishing touches to this column I thought I would ask Marcel for a comment on it.
I wrote:
I am just putting the finishing touches to BoW for the next magazine and I mention Vista at the end. Any idea when the new QPC2 will be ready for it or any comments you would like me to quote about it?
Oh, it isn't released yet? Then I should probably do that, thanks for reminding me ;-)
I have mentioned that, since QPC2 came out, this is the first time it has not been able to run on a new version of Windows without a change and that the problem was that M$ removed a DLL from its networking library. Anything else you would like to add?
They removed a function from one of their networking DLLs. This function was never officially documented, but I used it nonetheless. So in principle I am to blame, but considering to what great lengths they to go to keep everything as compatible as possible I was a bit amazed this actually happened.
So now you know - as always a new version of QPC2 should be with you before you know you need it!

Vista Startup

One last thing before I go. Vista is also trumpeting that you can now use Flash Ram as part of its operating memory. Well, well - we have had that on our QLs for ages. It is called ROMDisq. Time for Tony to sue Microsoft for intellectual property rights.

RWAP Services NEWS!

**** We have moved ****

See our updated address details below.

We have also acquired more brand new Sinclair QL membranes and another stock of Epson Stylus Colour 850 inkjet printers, so if you need a better printer for your QL, give us a shout.

More news is always available on our website: www.rwapsoftware.co.uk

We are also looking to produce some new hard disk interfaces for the ZX Spectrum and have a few little projects on the drawing board.

Our websites:
http://www.rwapservices.co.uk (General site)
http://www.rwapsoftware.co.uk (Sinclair computer second hand and new items)
http://www.rwapadventures.com (Adventure Programs)
http://www.internetbusinessangels.com (Guidance on setting up online businesses).

New Products!

QWORD 1.0 NOW WITH DIGITAL SOUND ON QPC2!

The wait is now over! Q-Word version 1 is finally available!

Platforms:
QPC/QXL, Q40/Q60, Aurora (with SGC)

Prices:

All versions without P-Word	£20.00
All versions with P-Word	£30.00

Notes:
Q-Word DOES NOT require SMSQ/E with GD2 support -OR- SMSQ/E at all on the Aurora or Qx0 machines. It works on the highest colour depth everywhere regardless of Operating System.
The Aurora version is available on either HD or ED disk. For the latter add £1.00 to the price. ED version is uncompressed and can be run directly from the floppy. All other Floppy versions are compressed. QPC/QXL version comes on CD. Non CD versions DO NOW support digital sound on QPC2

TALENT COMPUTER SYSTEMS for **Windows**

For QLers that run Windows or with incompatible hardware for Talent Games, we now have re-released these adventures so that they can run on your Windows-equipped PC. No Emulator, floppies, microdrive backups etc. required, just a one-click install! Of course the full QL line is still available! (See side column)

Talent Games for Windows ea. £ 10.00
(Each Game includes a runtime installation of QLAY-2 by Jimmy Montesinos)

Games Currently Available from www.rwapadventures.com

The Lost Kingdom of Zkul
West
The Prawn
Return to Eden

Replacement Sinclair QL Keyboard Membranes

We always have a stock of brand new Keyboard Membranes (and keyboard parts) for the original Sinclair QL, so if you have some keys which no longer work, just give us a call.

Cost is only £18.50 plus £2.75 post and packing.

Second Hand Items - Huge Range Available

We stock a wide range of books, hardware and software for the Sinclair QL, Z88 and ZX Spectrum, including disk interfaces, memory expansion and microdrive cartridges. If there is anything you need - have a look at www.rwapsoftware.co.uk (or ring us with details of your requirements).

We are always happy to help.

Old Favourites!

Utilities

SBASIC / SuperBASIC Reference Manual on CD	£ 20.00
Sidewriter v1.08	£ 10.00
Landscape Printing (EPSON printers)	
ImageD v1.03	£ 10.00
3D object generator	
Q-Help v1.06	£ 10.00
Superbasic On-Screen help system	
Q-Index v1.05	£ 5.00
Keyword-to-topic finder	
ProForma ESC/P2 Drivers v1.04 for ProWeSs	£ 8.00
Printer Driver	

Applications

Flashback SE v2.03 (upgrade only)	£ 5.00
Database	
QL Cash Trader v3.7	£ 5.00
Accounting/Finance	
QL Payroll v3.5	£ 5.00
Accounting/Finance	
QL Genealogist v3.26	£ 20.00
Genealogy	
Genealogy for Windows	£ 50.00
QL Genealogist to Windows version upgrade	£ 25.00
QL Cosmos v2.04	£ 5.00
Planetarium	
Q-Route v2.00	£ 25.00
Route Finding	
Upgrade from v1.xx	£ 5.00
Britain map v1.11	£ 2.00
BIG Britain map (needs 2Mb) v2.03	£ 5.00
Various Britain Area maps (ask for details)	ea. £ 2.00
Ireland map v1.00	£ 5.00
Belgium map v1.01	£ 2.00
Catalonia map v1.03	£ 2.00
P-Word UK English Dictionary (500.000 words!)	£ 15.00
Dictionary	

Leisure

Return to Eden v3.08	£ 10.00
Adventure	
Nemesis MkII v2.03	£ 8.00
Adventure	
The Prawn v2.01	£ 8.00
Adventure	
Horrorday v3.1	£ 8.00
Adventure	
West v2.00	£ 5.00
Adventure	
The Lost Kingdom of Zkul v2.01	£ 5.00
Adventure	
All 6 games above	£ 25.00
D-Day MkII v3.04	£ 10.00
Strategy/War Simulation	
Grey Wolf v1.08	£ 8.00
Graphical Submarine Simulation	
War in the East MkII v1.24 (upgrade only)	£ 5.00
Strategy/War Simulation	
Open Golf v5.20	£ 8.00
Sports Simulation	
QuizMaster II v2.07	£ 5.00
Quiz	
Stone Raider II v2.00	£ 5.00
Arcade Game	
Hoverzone v1.2	£ 5.00
Arcade Game	
Deathstrike v1.5	£ 5.00
Arcade Game	
Flightdeck v1.0	£ 10.00
Flight Simulation	
All 6 games above (Open Golf, QuizMaster II, Stone Raider II, Hoverzone, Deathstrike and Flightdeck)	£ 28.00

Notes on Software requirements

The following programs have a minimum SGC card requirement: P-Word, Qword, Big Britain MAP for Q-Route

RWAP Services
3 Dale View Court, Fulford, Stoke-On-Trent, Staffordshire ST11 9BA TEL: (+44) 1782 398143 From the UK Dial: 01782 398143
Website: http://www.rwapsoftware.co.uk
Email: sales@rwapsoftware.co.uk

We Accept Payment using:

PayPal VERIFIED NOCHEX EMAIL MONEY PREFERRED PAYMENT (Cheques in £ sterling made payable to R. Mellor)

A personal Statement from the Editor

by Geoff Wicks

Recently QL Today received a bizarre email. It came from

anonymous.(user code deleted)@anonymousspeech.com and made serious allegations against the editor.

Anonymousspeech.com is a company based in Japan and claims to be the world's largest anonymous email provider. It boasts that as it is subject to Japanese law it makes it "extremely expensive and troublesome for foreign private parties to obtain information about our subscribers". Among suggestions for use of the service are catching a cheating spouse; bypassing your banned email address; confirming suspicions about a friend or loved one; and performing checks as an employer or potential employee.

When a person hides behind anonymity, then others are entitled to speculate about his personality and motives. At QL Today we wondered if this was a genuine email, but decided on balance it was.

Normally as a matter of principle QL Today would not publish anonymous contributions, but we felt we should share the content of the email with our readers and give them a chance to comment. We wanted to publish the entire email, but faced legal complications. The way in which the email came to us made it uncertain whether it should be seen as a confidential private mailing or as a publishable public mailing. Normally we solve this type of problem by asking the author's permission to publish, but in this case the writer said that he did not want QL Today, or for that matter, anyone to contact him.

We are only able to report the gist of the email. The writer said he was a long time reader of QL Today, praised it as being of high technical quality and interesting, but would not be renewing his subscription. What he did not like was what he saw as the constant sniping against Quanta and, in effect, accused me as editor of using the magazine as a personal ego trip.

At this stage I should perhaps write something about my background which affects the way I handle differences of opinion. For about half of my working life I was in employment where it was my job to confront people about their behaviour. As a consequence I have been assaulted on several occasions, physically threatened on many more and verbally abused almost daily. The rewards came in seeing the transformation I achieved in my client's lives. This background means that I have a more robust attitude to conflict than many other people. I thump hard, but am also prepared to take hard thumps back.

When I became editor of QL Today I knew one of the difficult areas would be reporting Quanta and resolved I would do this critically, but in the true sense of the word. I would hit hard when it was justified and praise hard when it was deserved. This I have done.

Recently there has been a spate of challenging stories for Quanta in the magazine, but in one sense this has been the committee's own fault. When I published the first story about the error in the published constitution, the committee should have done the decent thing and apologise to the members. In so doing they would have killed the story. They did not do the decent thing, but chose to brazen it out and thus kept the story alive. I continued to investigate the constitution story and other matters came to light. One of these has been the unprecedented fall in Quanta membership and I think it healthy that this story is out of the closet. In three years Quanta membership has fallen by over 100. It is the most serious problem facing the committee, and has enormous implications for the future of Quanta, but try finding a mention of it in the 2007 AGM papers.

I have also praised hard. In this issue our lead news story is the Quanta Magazine. For years Quanta members have moaned about the poor quality of the magazine, but in the last 18 months John Gilpin and John Mason have achieved something quite remarkable. They have turned the Quanta Magazine back into a serious QL publication. Where have you seen public praise for their work? Many, many times in QL Today. Where else? On the QL-users group. Raised by whom? None other than your editor. Who gave his congratulations and thanks at the 2006 AGM? Once again your editor. How many other Quanta members have publicly thanked the Quanta committee for their transformation of the magazine? To my knowledge just one person, Per Witte. What a load of ungrateful people Quanta members are!

Our anonymous email writer also criticises my attitude to Quanta in the QL-users group. It is true that no one has criticised Quanta more than I have. Equally it is true that no one has praised Quanta more than I have. This year I have said on many an occasion that real gains are being made in Quanta; that Quanta is lucky to have an efficient secretary; that through his hard work another committee member has made significant improvements to Quanta; that Quanta appears now more willing to exploit its capital than previously; and that Quanta has been unfairly maligned by many people over its supposed failure to fund Goldfire.

I do not intend to change my reporting of Quanta. Indeed I am preparing another challenging story for the next issue. If readers do not like this, then our letter columns are open for your objections. And don't be afraid to hit hard. One of the great weaknesses of both QL Today and the Quanta Magazine is the lack of vigorous reader correspondence. However, if readers absolutely detest my reporting of Quanta, then it is time to find a new editor.

Just one final comment on Quanta. I am becoming increasingly convinced that the problem with Quanta is not people, but its structure. I know of no other organisation which gives its committee so much power and its members so little. A committee that is not challenged by its members tends to lose its negotiating skills and becomes inward looking and authoritarian. Equally when the members feel they have little real influence they become passive and apathetic. That is the state of Quanta today.

Changes Ahead

by Geoff Wicks

Changes are planned for QL Today.

Last time we reported on the vastly increased distribution costs we are facing, and readers will realise there are inevitable consequences for the future of QL Today. We were confronted with a choice of either raising the price or of slightly reducing the price and publishing the magazine less frequently.

The QL Today team have chosen for the latter option, and from Volume 12 we shall be publishing QL Today quarterly. In our opinion this will be the most practical way of reducing our production and distribution costs.

Where possible we publish the magazine to coincide with shows, but as the number of shows diminishes we expect to have to use expensive courier services more often to send the magazine from Germany to the UK.

At present we publish five times a year and find it difficult to maintain a regular schedule. No matter how well you plan, 5 issues cannot be easily spread over the 12 months of a year. A quarterly publication will make it easier for us to plan and hopefully become more reliable in our publishing schedule.

Another consideration has been the time the QL Today team must invest in the magazine. Although the editorial work can be fairly evenly spread over the year, with some peaking around the copy date, this is not true for the make-up and production. The average issue of QL Today has over three times the content of the average issue of the Quanta Magazine and this gives some idea of the huge time investment required in the month before publication. Quarterly publication would give some relief from the time pressures.

Editorially the magazine continues to remain viable and we are lucky to have a group of enthusiastic writers with different levels of QL use and experience. Our circulation also remains healthy, which indicates we are producing a magazine readers want and appreciate. We hope you will agree with us that quarterly publication is the best way of ensuring a good future for QL Today while giving our readers good value for their money.

Issue 5 of the current Volume 11 is planned to be ready mid/end of June, and from then on we plan to deliver Issue 1 of Volume 12 mid/end of September, Issue 2 mid/end of December, Issue 3 mid/end of March and issue 4 mid/end of June again.

Please help reducing the costs by returning the renewal form enclosed with this issue to save us having to send out reminders.

And - as always - please send any material for the next issue as soon as possible to help us getting the next issue out in time.

The QL Show Agenda

QL Meeting in Eindhoven

Saturady, 16th of June, 10:00 to 16:00
Pleincollege St. Joris, Roostenlaan 296

Thanks to the organiser, Sjef van de Molengraaf, the meetings at Eindhoven continue. Same venue as always. And for the March meeting, it was even the same, large room - not the one announcd in issue 2. J-M-S will be there, as always - and we hope to have the next issue of QL Today - issue 5 - ready for you to pick it up... and re-subscribe if you have not done it already by then.
Roy Wood of QBranch plans to attend to either this show or the one below, in October. Maybe this is an opportunity for a larger, international show. If you would like to come to a larger event, but not to a smaller one, then please let us know, which one you would prefer.

QL Meeting in Eindhoven

Saturady, 20th of October, 10:00 to 16:00
Pleincollege St. Joris, Roostenlaan 296

Same venue as always. J-M-S will be there, as always.

The Next Volume

Please fill in and return the renewal form enclosed as soon as possible, to help in saving costs.

German subscribers with automatic renewal and account debit do not need to return anything, the subscription will be automaticall renewed as agreed.

The Next Issue

We plan to have the next issue ready for you at Eindhoven, middle of June.
As always, it depends on how quickly we will get reviews, articles etc.
Some articles had to wait for the next issue ... I think the current issue with 64 pages holds the record so far. But we need more material, as always. The more we get and the sooner we get it, the quicker the next issue will be in your hands, and the better it will be.